D

NAVAL POSTGRADUATE SCHOOL
Monterey, California

THESIS

UNIFORM FRAMEWORK FOR GPS/IMU INTEGRATION USING KALMAN FILTERING

by

John B. Herrington

June, 1995

Thesis Advisor: Isaac I. Kaminer

Approved for public release; distribution is unlimited.

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

1. AGENCY USE ONLY *(Leave blank)*

2. REPORT DATE
June, 1995

3. REPORT TYPE AND DATES COVERED
Master's Thesis

4. TITLE AND SUBTITLE
UNIFORM FRAMEWORK FOR GPS/IMU INTEGRATION USING KALMAN FILTERING

5. FUNDING NUMBERS

6. AUTHOR(S)
LCDR John B. Herrington

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)
Naval Postgraduate School
Monterey, CA 93943

8. PERFORMING ORGANIZATION REPORT NUMBER

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)

10. SPONSORING/MONITORING AGENCY REPORT NUMBER

11. SUPPLEMENTARY NOTES
The views expressed in this thesis are those of the author and do not reflect the official policy or position of the Department of Defense or the United States Government.

12a. DISTRIBUTION/AVAILABILITY STATEMENT
Approved for public release; distribution is unlimited.

12b. DISTRIBUTION CODE

13. ABSTRACT *(Maximum 200 words)*

Today's modern avionics systems rely heavily on the integration of Global Positioning System (GPS) data and the air vehicle's accelerations obtained by an Inertial Measurement Unit (IMU). To properly resolve the GPS and the IMU data, one must have an understanding of the different coordinate systems involved. Since the GPS provides data in one coordinate reference frame and the IMU measures accelerations in another, transforming the data freely from one frame to the next is imperative if the avionics system is to provide meaningful data to the aircrew. This thesis provides a uniform approach to analysis and design of an integrated GPS/IMU avionics system using MATLAB/*Simulink* software development tools. Topics covered include: Coordinate Systems and Transformations ,Fundamentals of Inertial Sensors, Tangent Plane Navigation, Kinematic Equations and Error Analysis, Global Positioning System (GPS) Sources and Errors, and Kalman Filter Design. Emphasis is placed on addressing the theory and providing detailed examples to support each topic.

14. SUBJECT TERMS
GPS/IMU Integration, MATLAB, Simulink, Kalman Filtering, Coordinate systems, Coordinate transformations

15. NUMBER OF PAGES
234

16. PRICE CODE

17. SECURITY CLASSIFICATION OF REPORT
UNCLASSIFIED

18. SECURITY CLASSIFICATION OF THIS PAGE
UNCLASSIFIED

19. SECURITY CLASSIFICATION OF ABSTRACT
UNCLASSIFIED

20. LIMITATION OF ABSTRACT
UL

UNIFORM FRAMEWORK FOR GPS/IMU INTEGRATION USING KALMAN FILTERING

John B. Herrington
Lieutenant Commander, United States Navy
B.S., University of Colorado, Colorado Springs, 1983

Submitted in partial fulfillment of the
requirements for the degree of

MASTER OF SCIENCE IN AERONAUTICAL ENGINEERING

from the

NAVAL POSTGRADUATE SCHOOL

June, 1995

Author: John B Herrington
John B. Herrington

Approved by: I Kaminer
Isaac I. Kaminer, Thesis Advisor

Daniel J Collins
Daniel J. Collins, Second Reader

Daniel J Collins
Daniel J. Collins, Chairman
Department of Aeronautics and Astronautics

ABSTRACT

Today's modern avionics systems rely heavily on the integration of Global Positioning System (GPS) data and the air vehicle's accelerations obtained by an Inertial Measurement Unit (IMU). To properly resolve the GPS and the IMU data, one must have an understanding of the different coordinate systems involved. Since the GPS provides data in one coordinate reference frame and the IMU measures accelerations in another, transforming the data freely from one frame to the next is imperative if the avionics system is to provide meaningful data to the aircrew.

This thesis provides a uniform approach to analysis and design of an integrated GPS/IMU avionics system using MATLAB/*Simulink* software development tools. Topics covered include:

- Coordinate Systems and Transformations
- Fundamentals of Inertial Sensors
- Tangent Plane Navigation
- Kinematic Equations and Error Analysis
- Global Positioning System (GPS) Sources and Errors
- Kalman Filter Design

Emphasis is placed on addressing the theory and providing detailed examples to support each topic.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

ACKNOWLEDGMENT

Many thanks to Prof. Kaminer for his knowledge of, guidance with, and enthusiasm for the subject material. I owe a debt of gratitude to my fellow officers in the Aero 611 curriculum, their wit and wisdom entertained and encouraged me throughout my time at NPS. Most importantly, without the support of my family; Debbie, Jessica and Amanda, this effort would not have been possible. Thanks for your love and patience!

I. INTRODUCTION

The research conducted in the Avionics lab at the Naval Postgraduate School (NPS) Aeronautical Engineering Department, centers around the flight control of Unmanned Autonomous Vehicles (UAV's). Today's battlefield has become increasingly hostile to manned airborne vehicles as weapons have become more accurate and lethal. In order to reduce the risk, while still providing forward reconnaissance, the military has increasingly turned toward remotely operated aircraft. Autonomous vehicles, allow the commander in the field the opportunity to survey his environment and employ the UAV as a targeting, weapons, or intelligence gathering platform. The reduced risk to the pilot and the decreased cost when compared to a current fleet aircraft, makes the deployment of UAV's a high priority in future battles.

In order to control the vehicle in a variety of battlefield environments, precise knowledge of it's velocity and position is essential. Using on-board inertial navigation systems, the vehicle's accelerations can be precisely measured and used by on-board navigational computers to calculate the vehicle's velocity and position in some particular reference frame. An inertial navigation system, alone, can provide accurate position information in the short term, but must be integrated with an additional source if precise positional data is to be maintained for the long term. The UAV's requirement to operate aboard ships or in rugged locations makes it essential that extremely accurate position information is calculated. To meet this requirement an inertial navigation system can be integrated with an Global Positioning System (GPS) receiver. This provides the vehicle's navigation system with an additional, highly accurate positioning data source.

The purpose of this thesis is to provide the framework for designing an integrated inertial measurement / GPS system. To understand the system integration as a whole, it is necessary to have a solid understanding of the reference frames in which both systems operate and how the angular and translational motion of the vehicle relative to these reference frames is calculated. In an effort to make the three dimensional (3–D) relationships and the associated mathematics more understandable, numerous 3–D graphical examples are provided.

A brief overview of current inertial sensors; ring laser gyros and fiber optic gyros, is included in order to understand how vehicle accelerations are measured and how they are related to the body of vehicle.

As with any system, the IMU errors are inherent in the IMU's computations. Detailed analysis of errors is essential if the IMU accuracy is to be fully realized [Ref. 1, p. 231]. Since the IMU is not a position fixing system, incorporating a GPS as the fixing source and using a Kalman filter to combine the two, provides the best estimate of the vehicle's velocity and position. Kalman filter design will be discussed and developed using MATLAB/*Simulink* software development tools.

II. COORDINATE SYSTEMS

To develop the relationship between GPS and the IMU, an understanding of each coordinate system is essential. In this chapter, eight coordinate frames will be discussed with particular attention paid to only four. They are the Earth–Centered–Inertial (ECI) coordinate system, the Earth–Centered–Earth–Fixed (ECEF) coordinate system, the tangent plane coordinate system and the Wander Azimuth coordinate system.

A. TRUE INERTIAL $\{I\}$

The True Inertial coordinate system is a set of mutually perpendicular axes that neither accelerate nor rotate with respect to some fixed point in space. Newton assumed there was a reference frame whose absolute motion was zero (fixed relative to the stars) [Ref. 1, p. 9], and it is in this reference frame where they are valid. However, Newton's laws of motion can also be applied to any reference frame as long as the proper coordinate transformations are applied.

B. EARTH–CENTERED–INERTIAL $\{i\}$

The Earth–Centered–Inertial (ECI) coordinate system is centered about the origin of the earth and maintains a fixed orientation with respect to some inertial reference in space. As the earth rotates the frame stays oriented with respect to this inertial reference. Even though the ECI frame is referenced to the earth and translates as the earth rotates about the sun, it's rotational motion relative to the fixed point is space is minimal and can be ignored. This allows the assumption that Newton's laws of motion apply here also. In the cases we will discuss, we will assume the ECI frame stays fixed and all other frames will rotate with respect to the ECI frame.

C. EARTH–CENTERED–EARTH–FIXED $\{e\}$

The Earth–Centered–Earth–Fixed coordinate system, is similar to the ECI system except that the frame itself is connected to earth, that is, it rotates with the earth. Every twenty–four hours the ECEF frame coincides with the ECI frame. The system origin, as the name implies, is centered at the Earth's origin. The X–axis is directed through the intersection of the Greenwich meridian, 0° longitude and the equator, 0° latitude. The Y–axis is directed from the origin to intersect the equator at 90° east longitude. The Z–axis is directed along the Earth's spin axis from the origin through the north pole at 90° north latitude. The ECEF frame is depicted in Figure 2.1. The earth–centered earth–fixed system is independent of the mathematical model of the earth's surface. However, both the geodetic and the local geodetic systems depend on

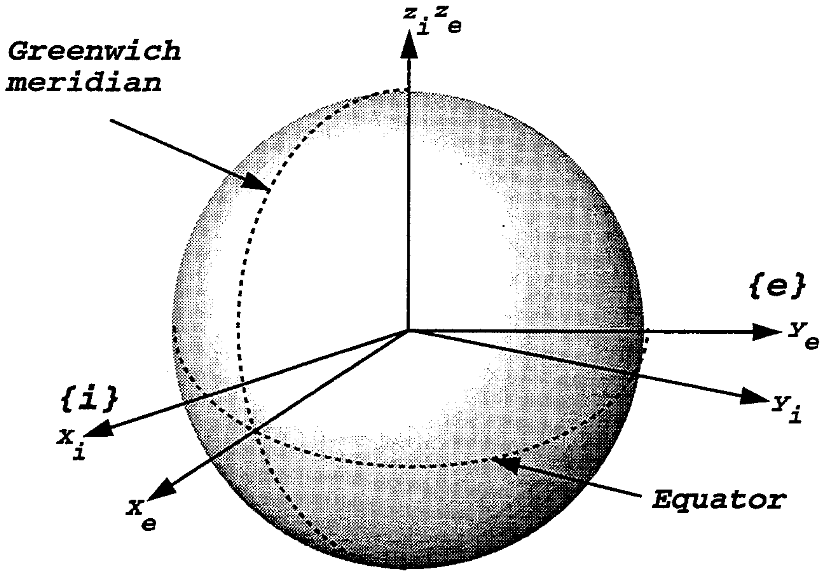


Figure 2.1: Earth–Centered, Earth–fixed Coordinate System

the specification of the earth model. The current standard for modeling the surface of the earth is the WGS–84 ellipsoid. This ellipsoid is generated by rotating an ellipse, whose semi–major axis is 6378137.0 meters and whose semi–minor axis is 6356752.3 meters, about its minor axis. The resulting closed surface is the model of the earth's surface. The true north pole (conventional terrestrial pole) and true south pole are the endpoints of the minor axis of the ellipsoid.

D. GEODITIC COORDINATE SYSTEM

The output of navigation systems used on aircraft today is generally latitude, longitude, and altitude — *i.e.* resolved in the geodetic coordinate system. This is the system used for describing positions of most earth bound objects. Charts developed for long range land and sea navigation invariably use geodetic coordinates.

The geodetic coordinate system is somewhat analogous to spherical coordinate system. The primary difference is that the elevation angle or latitude, ϕ, is the angle between the *ellipsoidal normal* and the equatorial plane. This means that the ray that defines this angle does not intersect the equatorial plane at the exact center of the earth. Instead, it intersects the equatorial plane at a small radius outside of the center as shown in Figure 2.2.

The longitude, λ, is identical to the spherical concept of that angle. It is the angle in the equatorial plane from 0° latitude and 0° longitude to any given point. Finally, h, the geodetic height or altitude, is the distance along the ellipsoidal normal away from the surface of the earth.

E. TANGENT PLANE COORDINATE SYSTEM

Typically, pure inertial systems navigate in a so–called tangent plane coordinate system, before outputting position in geodetic coordinates. The tangent plane system is defined by passing a plane at any point on the earth's surface. The intersection of

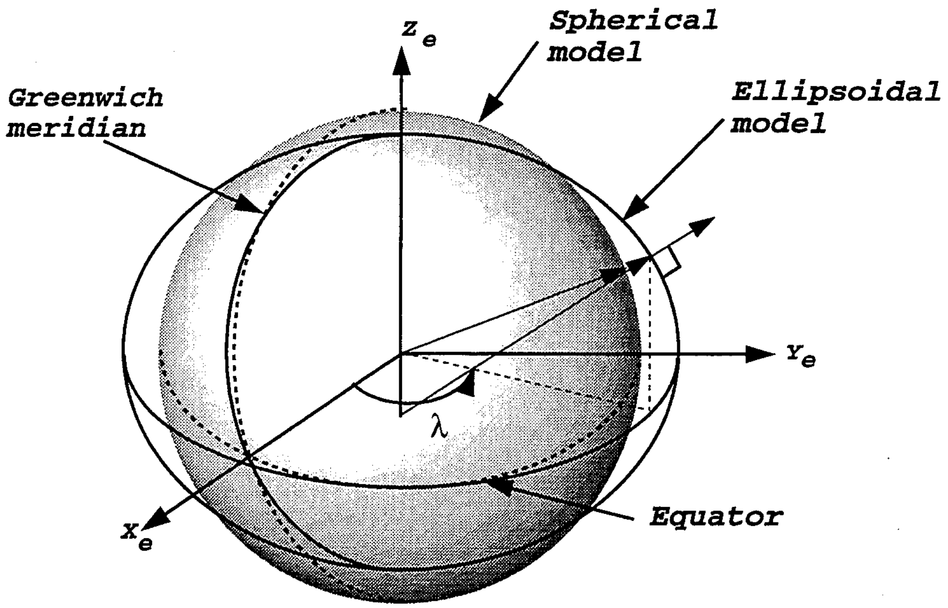


Figure 2.2: Geodetic Coordinate System

the plane with the surface of the earth becomes the origin of the system. The x–axis points toward true east. The y–axis points toward true north. Lastly, the z–axis is perpendicular to the defining plane of the system, away from the center of the earth. It is the z coordinate of the triad which defines a point's altitude in this system. This frame is sometimes referred to as the universal $\{u\}$ frame. This is shown in Figure 2.3.

F. NAVIGATION COORDINATE SYSTEM $\{n\}$

The navigation coordinate system is attached to the aircraft's frame. It's origin is located at the center of the aircraft's inertial navigation system [Ref. 1]. It maintains a local–level orientation to the reference ellipsoid in the same manner as the tangent plane but the orientation of the axes can be chosen at random. If the origin of the aircraft is co–located at the origin of the tangent plane, then the tangent plane and the navigation plane are one in the same. The frame shown in Figure 2.3, represents an East–North–Up (ENU) orientation where the x–axis points east, the

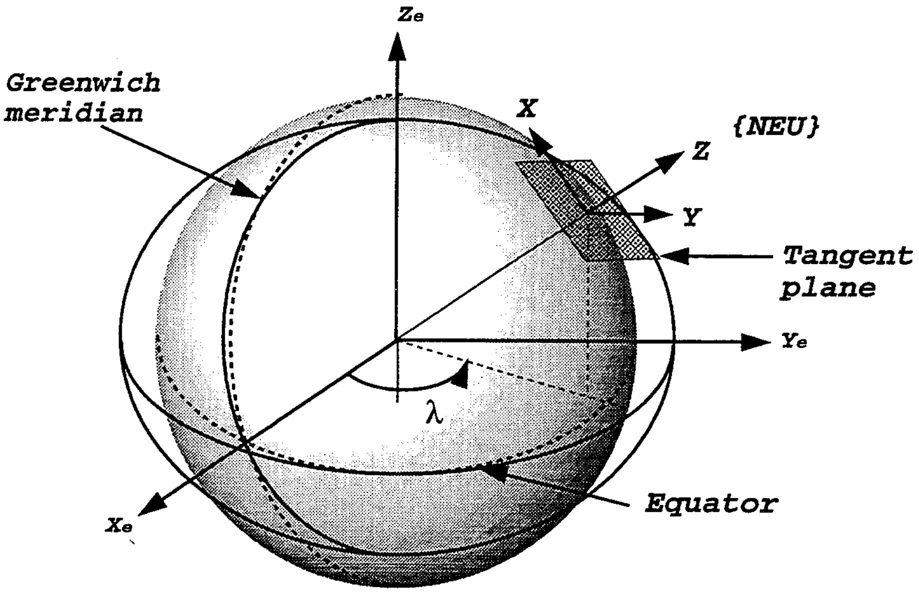


Figure 2.3: Tangent Plane Coordinate System

y–axis points north and the z– axis points up. The orientation of the frame can be specified any number of ways; North–East–Up (NEU) or North–East–Down (NED).

G. WANDER AZIMUTH COORDINATE SYSTEM $\{c\}$

The Wander Azimuth coordinate system is a frame centered at some reference point on the body, most likely the center of the aircraft's inertial system, the origin of the navigation frame. The x and y plane is tangent to the local vertical with the z axis perpendicular. The frame is defined with respect to the ECEF frame using longitude λ, latitude ϕ and the wander azimuth angle α. The angle α is positive when the frame is to the west of true north, the angle λ is positive east of the Greenwich meridian, and ϕ is positive north of the equatorial plane. If the wander azimuth frame is aligned with the ECEF frame at the Greenwich meridian, the angle α is zero. The Wander Azimuth, co–located tangent and navigation planes and the ECEF frames are depicted in Figure 2.4.

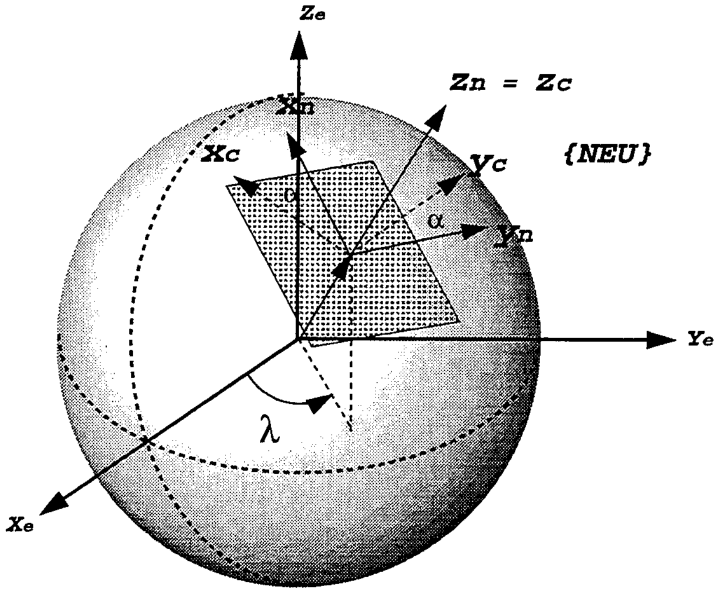


Figure 2.4: Wander Azimuth, Navigation and ECEF Coordinate Systems

H. PLATFORM FRAME $\{p\}$

The platform coordinate system is the right-handed orthogonal frame defined by the input axes of the inertial sensors. If the inertial navigation device is a "strapdown" model, the aircraft's accelerations are measured in the platform frame. Imagine the inertial system, perhaps a ring laser gyro, bolted to the frame of the aircraft and all accelerations experienced by the aircraft are directly measured by the sensors of the inertial unit. Platform accelerations are equivalent to body fixed accelerations if the unit is a "strapdown system". In this case the platform frame would be referred to as the body fixed frame $\{b\}$. This assumption will be valid throughout this thesis.

I. ACCELEROMETER FRAME $\{a\}$

The accelerometer frame is a non-orthogonal frame defined by the sensitive axes of the accelerometers which make up the inertial navigation system [Ref. 2]. If the axes of the inertial measurement unit are not properly aligned with the platform, errors in acceleration measurements can exist [Ref. 1].

III. COORDINATE TRANSFORMATIONS

The kinematic equations that relate the motion of a body relative to a fixed reference frame must be given in terms of the relative angular velocity and coordinate transformations between each frame. In order to use these coordinate systems, one must be able to transform between them freely. For example, satellite positions for the GPS computations are given in ECEF coordinates. Suppose the aircraft position is wanted in the tangent plane coordinate system. Therefore, a transformation between ECEF and tangent plane systems is required. For the three systems above, six conversions are required. By being able to convert from geodetic to ECEF, and ECEF to local geodetic, one can convert from geodetic to local geodetic by chaining the two conversions together. Thus, the only four transformations discussed are:

- geodetic to ECEF
- ECEF to geodetic
- ECEF to tangent plane
- tangent plane to ECEF

A. EULER ANGLES

Euler angles are used to define the orientation of two coordinate systems with respect to each other. For example, consider the aircraft body fixed coordinate system $\{b\}$ and how it is oriented with respect to the tangent plane or universal frame $\{u\}$. We define the angles the aircraft's body makes with this system in terms of roll ϕ, pitch θ, and yaw ψ. Written in vector format:

$$\Lambda = \begin{bmatrix} \phi \\ \theta \\ \psi \end{bmatrix} \tag{3.1}$$

Figure 3.1 represents body and universal coodinate frames where the origins are collocated.

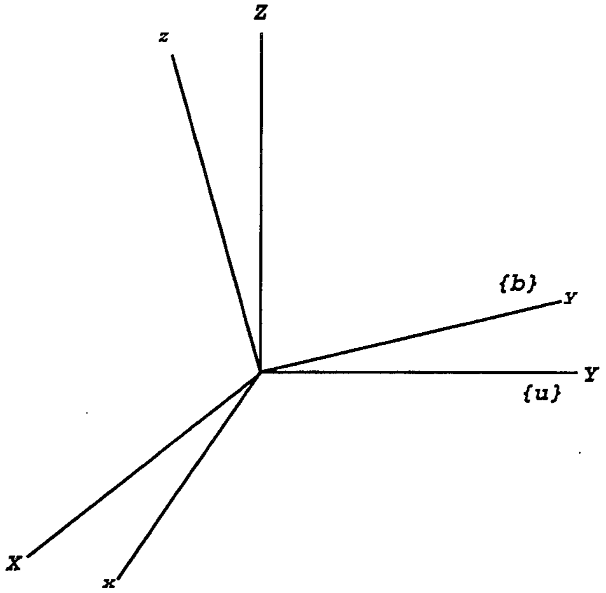


Figure 3.1: Collocated Universal $\{u\}$ and Body $\{b\}$ Coordinate Systems

Given a vector y, resolved in $\{u\}$: $y_u = \begin{bmatrix} x \\ y \\ z \end{bmatrix}$, what are the components of y in $\{b\}$ or y_b? This can be accomplished using Euler angles as follows. First, let's rotate the $\{u\}$ frame about the z-axis. Since the angle ψ describes rotation about the z axis, our first rotation matrix will be a function of ψ. Figure 3.2 depicts the rotation about the z-axis. The z-axis should be considered positive outward from the x-y plane. The rotation is considered positive using the right-hand-rule.

The new axes of the rotated frame have been labeled (x', y', z'). Since the transformation matrix describes the trigonometric relationships between the old and new frames let's use Figure 3.3 to determine what these relationships are.

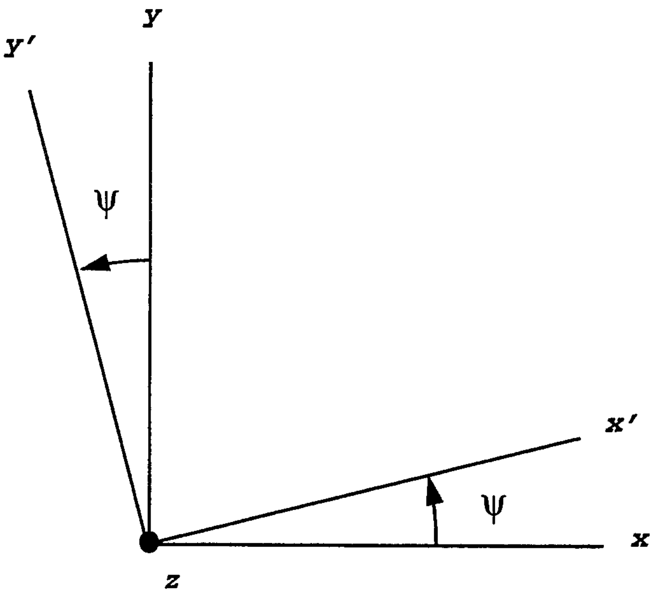


Figure 3.2: Rotation About the z Axis

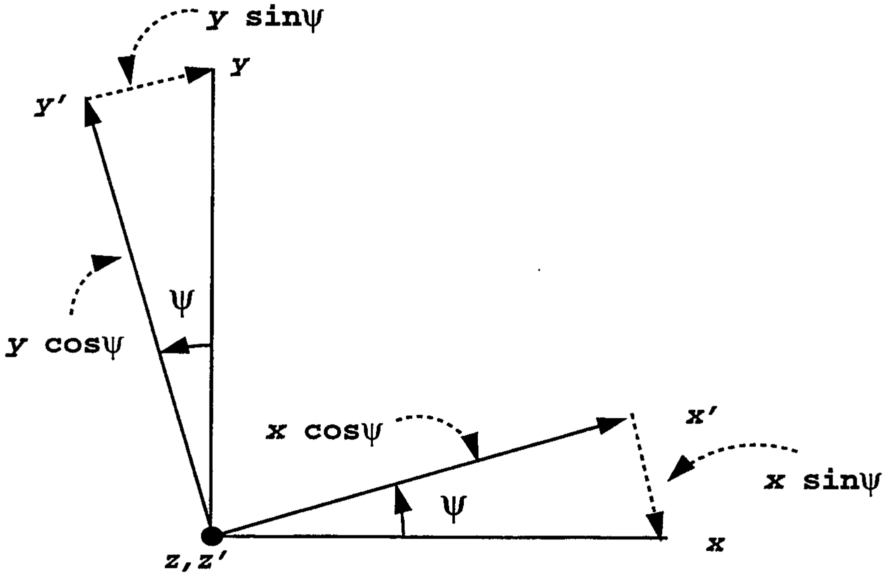


Figure 3.3: Trigonometric Relationships Between Rotated Frames

We see x' consists of two components. One, is the projection of x onto x' or $x\cos\psi$. The other is the $y\sin\psi$. Writing x' as a function of x and y we get:

$$x' = x\cos\psi + y\sin\psi \tag{3.2}$$

The same relationship exists for y'. In this case the $x \sin \psi$ component is in the direction opposite of our sign convention. Therefore y' written as a function of the components of x and y becomes:

$$y' = -x \sin \psi + y \cos \psi \tag{3.3}$$

Since the rotation occurred about the z-axis, $z' = z$. Gathering the terms in matrix format we have the following transformation matrix:

$$\begin{bmatrix} x' \\ y' \\ z' \end{bmatrix} = \begin{bmatrix} \cos\psi & \sin\psi & 0 \\ -\sin\psi & \cos\psi & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \end{bmatrix}. \tag{3.4}$$

Let's now perform the rotation about the new y' axis. This time the rotation angle is θ and the resulting axes will be (x'', y'', z''). Figure 3.4 shows the relationship between the new axes.

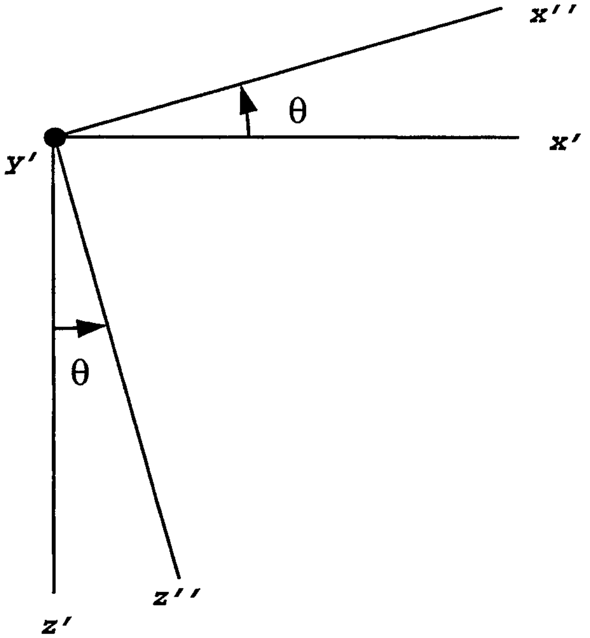


Figure 3.4: Rotation About the y' Axis

Using the same process we used to calculate (x', y', z'), we develop a transformation matrix based on the trigonometric relationships between the new axes. These relationships are shown in Figure 3.5.

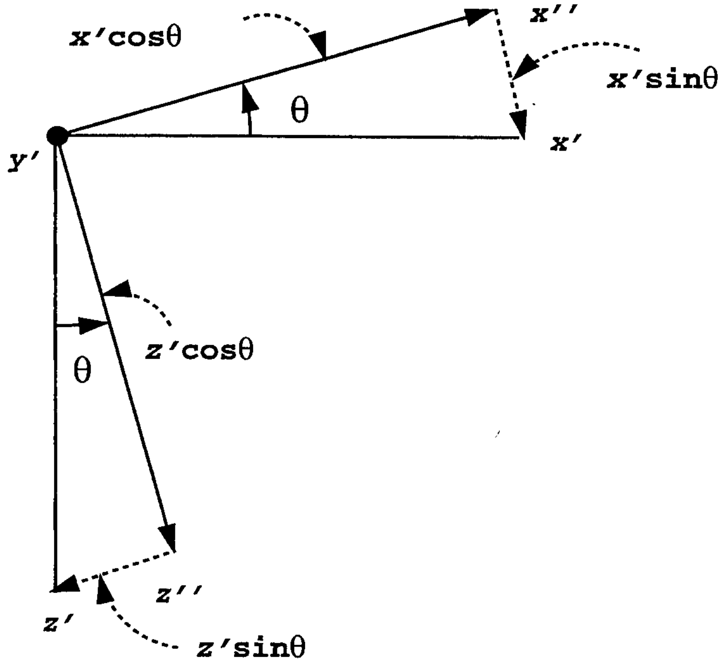


Figure 3.5: Trigonometric Relationships Between Rotated Frames

The new transformation matrix becomes:

$$\begin{bmatrix} x'' \\ y'' \\ z'' \end{bmatrix} = \begin{bmatrix} \cos\theta & 0 & -\sin\theta \\ 0 & 1 & 0 \\ \sin\theta & 0 & \cos\theta \end{bmatrix} \begin{bmatrix} x' \\ y' \\ z' \end{bmatrix}. \tag{3.5}$$

Notice that the negative sign now appears in the first row with the sin term. This is due to the fact that the rotation was through the angle θ in the positive direction, or nose up. The x'' component, $-z' \sin\theta$, was in the direction opposite of the sign convention chosen.

The last transformation will be about the x'' axis which is a rotation of ϕ, right wing down. This is the final orientation of the body frame and results in a transformation matrix:

$$\begin{bmatrix} x''' \\ y''' \\ z''' \end{bmatrix} = \begin{bmatrix} 0 & 0 & 1 \\ 0 & \cos\phi & \sin\phi \\ 0 & -\sin\phi & \cos\phi \end{bmatrix} \begin{bmatrix} x'' \\ y'' \\ z'' \end{bmatrix}. \tag{3.6}$$

Combining equations 3.4, 3.5, and 3.6, we have the final (3–2–1) transformation from (x, y, z) to (x_b, y_b, z_b) as:

$$\begin{bmatrix} x \\ y \\ z \end{bmatrix}_b = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\phi & \sin\phi \\ 0 & -\sin\phi & \cos\phi \end{bmatrix} \begin{bmatrix} \cos\theta & 0 & -\sin\theta \\ 0 & 1 & 0 \\ \sin\theta & 0 & \cos\theta \end{bmatrix} \begin{bmatrix} \cos\psi & \sin\psi & 0 \\ -\sin\psi & \cos\psi & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \end{bmatrix} \tag{3.7}$$

When performing transformations, the Euler angles associated with each rotation are not vectors. That is, $\theta + \phi \neq \phi + \theta$. This implies that the order of rotation is important. In the previous example, the rotation order was 3–2–1, or ψ, θ, ϕ. To verify, perform a 2–3–1, or ψ, ϕ, θ and compare the final transformation matrix with equation 3.7.

B. GEODITIC TO ECEF COORDINATE TRANSFORMATION

In order to compute the transformation from geodetic to ECEF coordinates, three auxiliary quantities — f, e, and N — must first be defined [Ref. 5]. The flattening factor, f, represents the relative flatness of the ellipsoid. A zero flattening factor would mean an unflattened ellipse (a sphere), while a unity value would mean a totally flattened ellipsoid (a circle in the plane perpendicular to the minor axis). The mathematical definition of f is

$$f = \frac{a - b}{a}, \tag{3.8}$$

where a and b are the semi–major and semi–minor axes of the ellipsoid, respectively.

Directly related to the flattening factor is the eccentricity, ε. It is defined by

$$\varepsilon^2 = 2f - f^2. \tag{3.9}$$

The eccentricity is a variable similar to the flattening factor. It represents how close the ellipsoid is to a sphere. It, too, is zero for a sphere and one for a completely flat figure. The eccentricity, rather than the flattening factor, is typically used in coordinate transformations.

Lastly, N is the length of the ellipsoidal normal from the ellipsoidal surface to its intersection with the ECEF z-axis. Mathematically, N is

$$N = \frac{a}{\sqrt{1-\varepsilon^2 \sin^2 \phi}}, \tag{3.10}$$

where ϕ is the geodetic latitude.

Using these quantities, one may define the transformation

$$\begin{aligned} x &= (N+h)\cos\phi \, \cos\lambda \\ y &= (N+h)\cos\phi \, \sin\lambda \\ z &= [N(1-\varepsilon^2)+h]\sin\phi. \end{aligned} \tag{3.11}$$

C. ECEF TO GEODITIC COORDINATE TRANSFORMATION

The transformation from ECEF to geodetic coordinates is clearly the inverse of the process presented in the previous section. First, λ, the longitude, can be found by dividing first two expressions of Equations 3.11 yielding

$$\tan\lambda = \frac{y}{x}. \tag{3.12}$$

By examining the geometry in Figure 2.2, one can determine the following relationship

$$\tan\phi = \frac{(N+h)\sin\phi}{\sqrt{x^2+y^2}}, \tag{3.13}$$

which is a non-linear equation in ϕ. Solving the third of Equations 3.11 for $(N+h)\sin\phi$ and substituting into Equation 3.13 yields

$$\tan\phi = \frac{z}{\sqrt{x^2+y^2}}(1+\frac{\varepsilon^2 N \sin\phi}{z}), \tag{3.14}$$

which is still an analytically unsolvable equation in geodetic latitude. To solve this equation, one initially assumes that h is zero, an excellent assumption for intra–atmospheric flight. Now, the z equation in Equation 3.11 can be simplified to

$$z_{h=0} = N(1 - \varepsilon^2)\sin\phi. \tag{3.15}$$

This equation can be substituted into Equation 3.14 to give the initial solution for ϕ

$$\tan\phi_1 = \frac{1}{1 - \varepsilon^2}\frac{w}{\sqrt{x^2 + y^2}}. \tag{3.16}$$

This initial solution for ϕ can be substituted back into the second term of Equation 3.14 to yield an updated ϕ. Iteration of this process commencing with Equation 3.14 continues until the geodetic latitude stops changing. Finally, solving Equation 3.13 for h

$$h = \frac{\sqrt{x^2 + y^2}}{\cos\phi} - N, \tag{3.17}$$

which completes the conversion.

If we assume a spherical earth model, a and b in Equation 3.8 are equal, resulting in a flattening factor $f = 0$. (Assume $a = b = R_o$, radius of the earth). Substituting f in Equation 3.9, the eccentricity factor e also equals zero. Applying these values to Equation 3.10, the length of the ellipsoidal normal N is now simply the radius of the earth R_o. Substituting these values in equations 3.11 through 3.18, the altitude h can now be defined as

$$h = \frac{\sqrt{x^2 + y^2}}{\cos\phi} - R_o \tag{3.18}$$

The physical representation can be seen in Figure 3.6.

D. ECEF TO TANGENT PLANE COORDINATE TRANSFORMATION

Anytime one uses a local geodetic or tangent plane coordinate system, one must first specify the geodetic coordinates — latitude and longitude – of the origin. Once

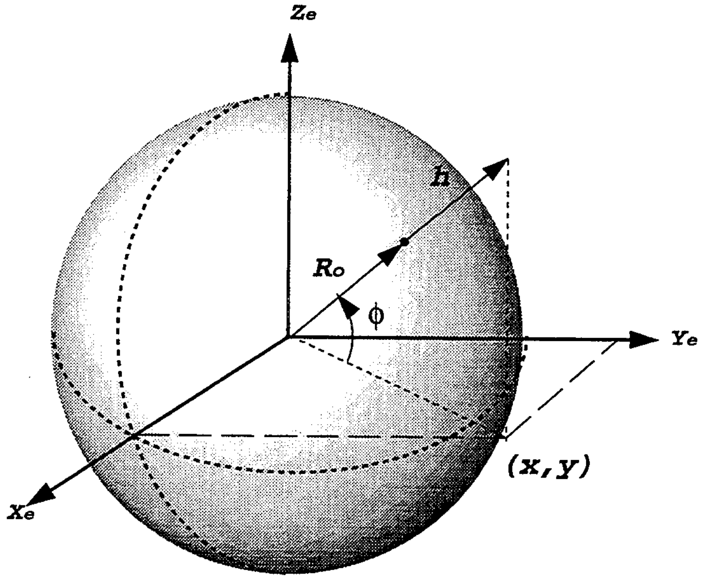


Figure 3.6: ECEF Spherical Model with Altitude

specified, the origin must be expressed in ECEF coordinates. Then, a vector from this origin to the point being transformed can be formed, resolved in the ECEF system

$$\begin{bmatrix} \Delta x \\ \Delta y \\ \Delta z \end{bmatrix}_e = \begin{bmatrix} x_2 - x_1 \\ y_2 - y_1 \\ z_2 - z_1 \end{bmatrix}, \tag{3.19}$$

where the '2' subscript denotes the point being transformed. The product of two rotation matrices operates on the difference vector defined in Equation 3.19 to yield the local geodetic coordinates of the point P_u, where the subscript $\{u\}$ represents the universal frame.

$$P_u = \begin{bmatrix} x \\ y \\ z \end{bmatrix}_u = \begin{bmatrix} -\sin\lambda & \cos\lambda & 0 \\ -\sin\phi\,\cos\lambda & -\sin\phi\,\sin\lambda & \cos\phi \\ \cos\phi\,\cos\lambda & \cos\phi\,\sin\lambda & \sin\phi \end{bmatrix} \begin{bmatrix} \Delta x \\ \Delta y \\ \Delta z \end{bmatrix}_e, \tag{3.20}$$

where λ is the geodetic longitude and ϕ is the geodetic latitude.

E. TANGENT PLANE TO ECEF COORDINATE TRANSFORMATION

The transformation from tangent plane to ECEF can be derived by merely reversing the process developed in the previous section.

First, the origin of the local geodetic system must be converted from geodetic to ECEF coordinates. Next, the inverse of the rotations performed in Equation 3.20 must be executed yielding the Δ vector from the origin of the tangent plane system to the point being transformed. This vector, expressed in ECEF coordinates is

$$\begin{bmatrix} \Delta x \\ \Delta y \\ \Delta z \end{bmatrix}_e = \begin{bmatrix} -\sin\lambda & -\sin\phi\cos\lambda & \cos\phi\cos\lambda \\ \cos\lambda & -\sin\phi\sin\lambda & \cos\phi\sin\lambda \\ 0 & \cos\phi & \sin\phi \end{bmatrix} \begin{bmatrix} x \\ y \\ z \end{bmatrix}_u . \tag{3.21}$$

Clearly, adding the Δ vector to the position of the origin of the tangent plane system (both now in ECEF coordinates) completes the transformation:

$$\begin{bmatrix} x \\ y \\ z \end{bmatrix}_e = \begin{bmatrix} x_{orig} \\ y_{orig} \\ z_{orig} \end{bmatrix}_e + \begin{bmatrix} \Delta x \\ \Delta y \\ \Delta z \end{bmatrix}_e . \tag{3.22}$$

F. VECTOR NOMENCLATURE

In order to proceed further, there must be a firm understanding of the nomenclature used to define the vector of interest and the frame of reference where it is resolved. The vectors we will be interested in are position P, linear velocity V, and angular velocity ω. Each vector has a magnitude and direction and a reference frame to which it is attached. For the purpose of this discussion, we will be concerned only with two reference frames, the tangent (universal), denoted by u, and the body fixed frame, denoted by $\{b\}$. As we become more familiar with other reference frames we will use different characters.

Let Q be an arbitrary point located on the body of the aircraft. We can draw a position vector to that point from the origin of the body reference frame. This vector would be identified as ${}^{b}P_Q$. The P stands for a position vector, the subscript Q for the point of interest and the superscript b for the reference frame the point is measured with respect to. Simply put, ${}^{b}P_Q$ is the position of point Q resolved in $\{b\}$. As with any vector, the position vector can be described in terms of any reference frame. An example would be to define the point Q with respect to the universal frame or ${}^{u}P_{Q}$. Figure 3.7 illustrates the position vectors in both reference frames.

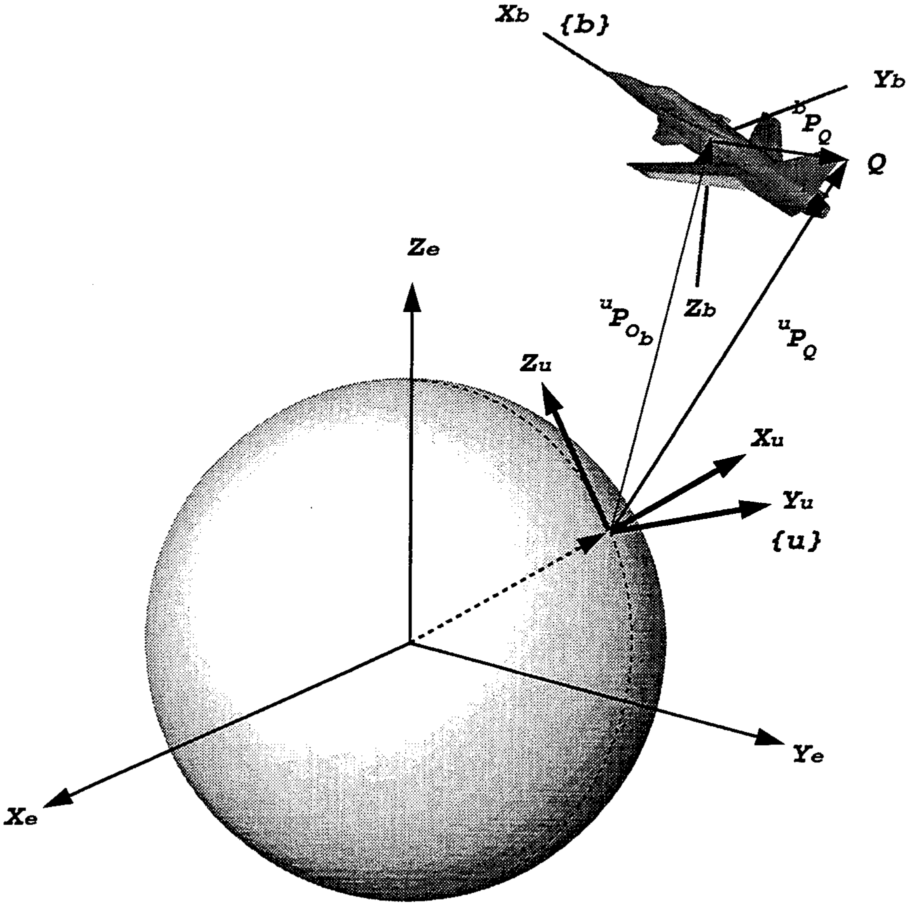


Figure 3.7: Position Vectors of Point Q

If the position vector P refers to the origin of a reference frame, the subscript will be o_x where o refers to the origin and x refers to the frame of interest. Therefore, ${}^{i}P_{o_b}$ would refer to the position of the origin of the $\{b\}$ frame, resolved in the universal frame $\{u\}$.

G. TRANSFORMATION OF ANGULAR VELOCITIES

Now that we can describe the position of a point on a body with respect to an arbitrary reference frame, we also need to express the motion of the body as well. As the vehicle moves in inertial space, the angles that describe it's relative position change as well. Question: How do we define the Euler angular rates in terms of the body's angular rates?

$$\dot{\Lambda} = \begin{bmatrix} \dot{\phi} \\ \dot{\theta} \\ \dot{\psi} \end{bmatrix} = ?\omega = ? \begin{bmatrix} p \\ q \\ r \end{bmatrix}$$

Answer: Let Ω_{bn} be the angular velocity of $\{b\}$ with respect to $\{n\}$. We can obtain $\dot{\Lambda}$ by considering the derivative of the Euler angle in the coordinate system it was defined. The total angular velocity is:

$$\Omega_{bn} = C \begin{bmatrix} 0 \\ 0 \\ \dot{\psi} \end{bmatrix} + DB \begin{bmatrix} 0 \\ \dot{\theta} \\ 0 \end{bmatrix} + D \begin{bmatrix} \dot{\phi} \\ 0 \\ 0 \end{bmatrix} \tag{3.23}$$

Where C is the transformation matrix from $\{n\} \to \{b\}$:

$$C = \underbrace{\begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\phi & \sin\phi \\ 0 & -\sin\phi & \cos\phi \end{bmatrix}}_{D} \underbrace{\begin{bmatrix} \cos\theta & 0 & -\sin\theta \\ 0 & 1 & 0 \\ \sin\theta & 0 & \cos\theta \end{bmatrix}}_{B} \begin{bmatrix} \cos\psi & \sin\psi & 0 \\ -\sin\psi & \cos\psi & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

DB is the transformation matrix from $\{n'\} \to \{b\}$, and D is the transformation matrix from $\{n''\} \to \{b\}$. Combining terms results in:

$$\omega = \begin{bmatrix} 1 & 0 & -\sin\theta \\ 0 & \cos\phi & \cos\theta\sin\phi \\ 0 & -\sin\phi & \cos\theta\cos\phi \end{bmatrix} \begin{bmatrix} \dot{\phi} \\ \dot{\theta} \\ \dot{\psi} \end{bmatrix}. \tag{3.24}$$

Finally, ${}^n_b\dot{C} = {}^n_bCS(\omega)$, where:

$$S(\omega) = \begin{bmatrix} 0 & -\omega_z & \omega_y \\ \omega_z & 0 & -\omega_x \\ -\omega_y & \omega_x & 0 \end{bmatrix} \tag{3.25}$$

is a skew symmetric matrix.

H. SPECIFIC TRANSFORMATIONS

1. ECI to ECEF: ${}^{e}_{i}C$

First, we need to determine Earth's sidereal rate, Ω_{ei} The subscript ei implies the angular rate of $\{e\}$ measured with respect to $\{i\}$. This convention is the reverse of the nomenclature in [Ref. 1] and will be maintained due to it's simplicity. Assume the navigation frame and universal frame are co–located as shown in Figure 3.8. Consider Figure 3.9. It is clear from this figure that Ω_{ei} resolved in $\{n\}$ is:

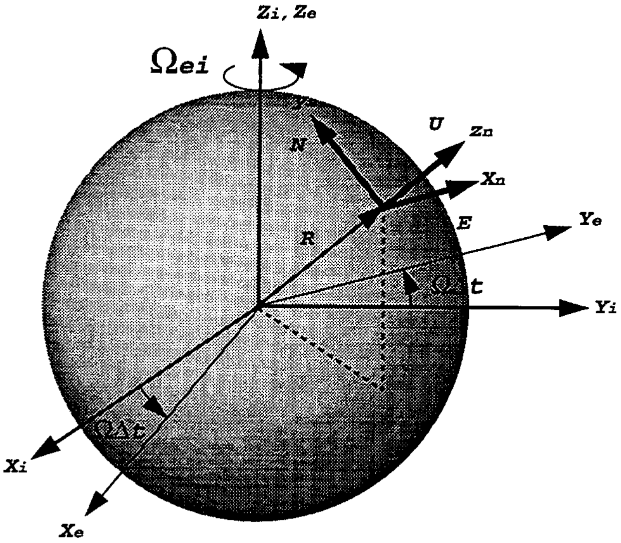


Figure 3.8: ENU Navigation Frame

$$ {}^{n}\Omega_{ei} = \begin{bmatrix} 0 \\ \Omega\cos\phi \\ \Omega\sin\phi \end{bmatrix}, \tag{3.26}$$

where Ω is the magnitude of Earth's rotation and is given by:

$$\Omega = \frac{360^\circ}{23 + \frac{\left[56 + \frac{4.09}{60}\right]}{60}}$$

$$\Omega = 15.041 \frac{deg}{hr} = 7.2 \times 10^{-5} \frac{rad}{sec}$$

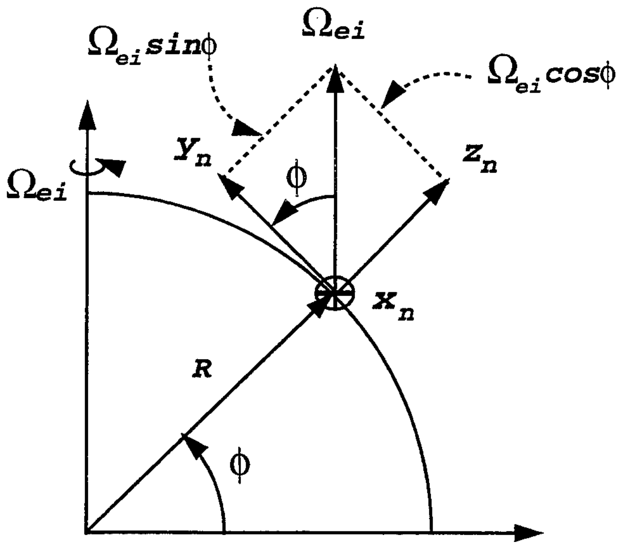


Figure 3.9: Earth's Sidereal Rate

where one sidereal day equals 23hr 56min 4.09sec.

Resolving the angular velocity in the ECEF $\{e\}$ frame:

$$^{e}\Omega_{ei} = \begin{bmatrix} 0 \\ 0 \\ \Omega \end{bmatrix} \tag{3.27}$$

This vector implies that the only component of Ω_{ei}, in the $\{e\}$ frame is in the z direction, see Figure 3.8.

Now, consider the rotation viewed in the equatorial plane, where $z_i = z_e$. Over a time Δt, the earth rotates at a rate $\Omega\Delta t$, figure 3.10. The transformation matrix from $\{i\}$ to $\{e\}$ can be calculated as:

$$\begin{bmatrix} x \\ y \\ z \end{bmatrix}_e = \begin{bmatrix} \cos\Omega\Delta t & \sin\Omega\Delta t & 0 \\ -\sin\Omega\Delta t & \cos\Omega\Delta t & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \end{bmatrix}_i \tag{3.28}$$

Let $\tilde{\Lambda} = \Omega(t - t_o) - 2n\pi$, where n is chosen such that $0 \leq \Lambda \leq 2\pi$ & $(t - t_o)$ is the time elapsed since vernal equinox. Since the two frames are orthogonal the transformation from $\{e\}$ to $\{i\}$ would be the inverse of the transformation matrix in Equation 3.28. Substituting Λ and inverting the matrix gives:

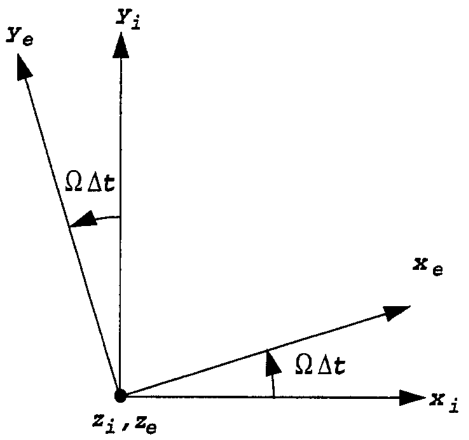


Figure 3.10: Sidereal Rate in the Equatorial Plane

$$\begin{bmatrix} x \\ y \\ z \end{bmatrix}_i = \begin{bmatrix} c\tilde{\Lambda} & -s\tilde{\Lambda} & 0 \\ s\tilde{\Lambda} & c\tilde{\Lambda} & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \end{bmatrix}_e \tag{3.29}$$

Finally, determine a point on the Earth's surface in terms of latitude and longitude (ϕ, λ). Let P be the vector describing the point Q, Figure 3.11, then:

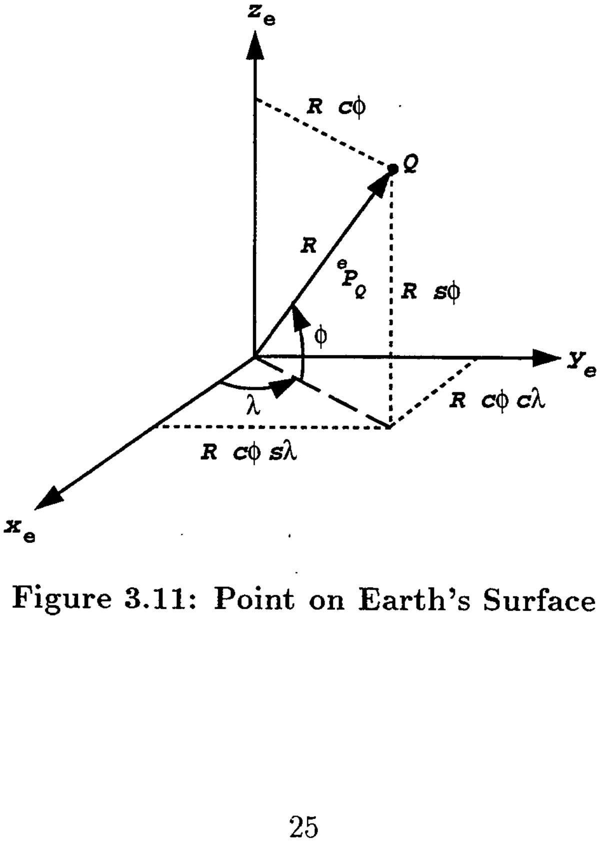


Figure 3.11: Point on Earth's Surface

$$
{}^{e}P = \begin{bmatrix} R\cos\phi\cos\lambda \\ R\cos\phi\sin\lambda \\ R\sin\phi \end{bmatrix} \tag{3.30}
$$

where R is the radius of the Earth.

2. ECEF to Navigation: ${}^{n}_{e}C$

Now consider the transformation from $\{e\}$ to $\{n\}$. Let the orientation of the navigation frame be:

$$x_n = U,\ y_n = E,\ z_n = N$$

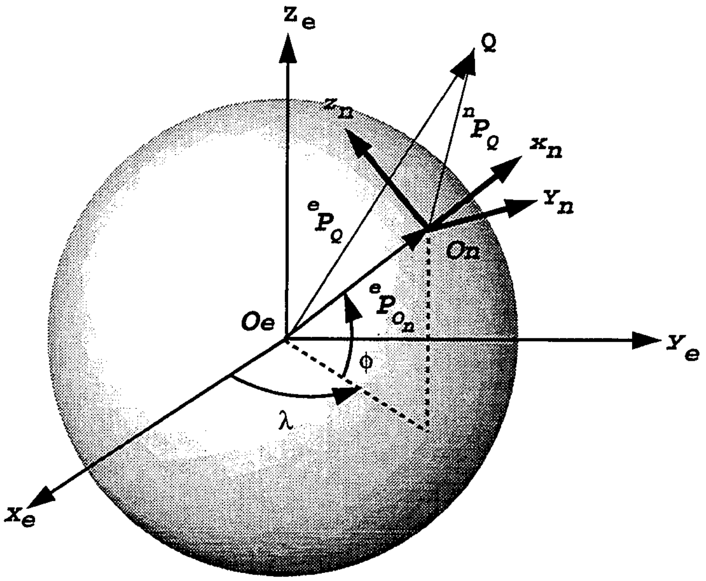


Figure 3.12: ECEF to Navigation

Then

- Rotating around z_e by λ (longitude)
- Rotating around y_e by ϕ (latitude)

to obtain:

$$^{n}P_Q = \begin{bmatrix} x \\ y \\ z \end{bmatrix}_n = \begin{bmatrix} c\phi & 0 & s\phi \\ 0 & 1 & 0 \\ -s\phi & 0 & c\phi \end{bmatrix} \begin{bmatrix} c\lambda & s\lambda & 0 \\ -s\lambda & c\lambda & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \end{bmatrix}_e = {}^{n}_{e}C\, {}^{e}P_Q \tag{3.31}$$

Note: Since the origins in Figure 3.12 are not coincident $\Rightarrow$ Eq. 3.31 is not completely correct. We need to include position of the origin of the navigation frame, ${}^{e}P_{o_n}$.

$${}^{e}P_Q = {}^{e}P_{o_n} + {}^{e}_{n}C\, {}^{n}P_Q \Rightarrow {}^{n}P_Q = {}^{n}_{e}C\,({}^{e}P_Q - {}^{e}P_{o_n})$$

Use vector calculus to define unit vectors for $\{n\}$:

- $\hat{\imath}_E = \frac{\Omega_{ei} \times R}{|\Omega_{ei} \times R|}$
- $\hat{\imath}_N = \frac{R \times (\Omega_{ei} \times R)}{|R \times (\Omega_{ei} \times R)|}$
- $\hat{\imath}_U = \frac{R}{|R|}$

3. ECEF to Wander Azimuth $\{c\}$: ${}^{c}_{e}C$

To obtain ${}^{c}_{n}C$ we must rotate $\{n\}$ by an angle α around x_n:

where:

$${}^{c}_{n}C = \begin{bmatrix} 1 & 0 & 0 \\ 0 & c\alpha & s\alpha \\ 0 & -s\alpha & c\alpha \end{bmatrix} \tag{3.32}$$

Then ${}^{c}_{e}C = {}^{c}_{n}C\, {}^{n}_{e}C$, see figure 3.13.

Now, for $\{n\} = NEU$

$${}^{c}_{n}C = \begin{bmatrix} c\alpha & -s\alpha & 0 \\ s\alpha & c\alpha & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{3.33}$$

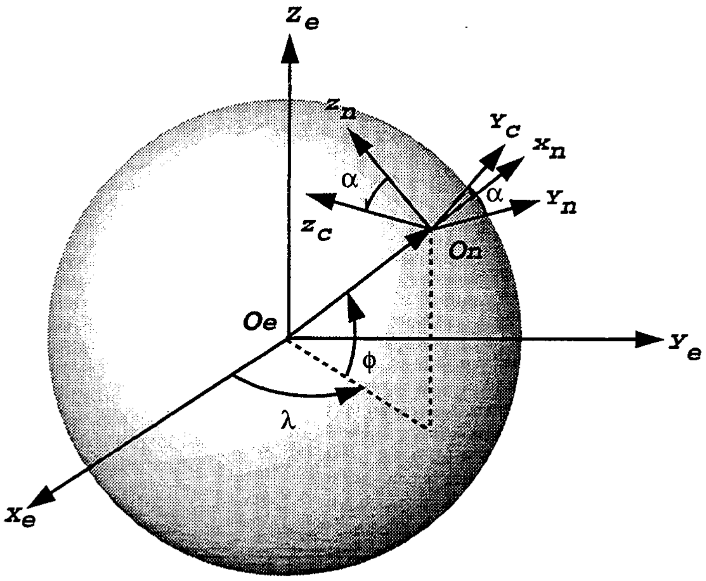


Figure 3.13: ECEF to Wander Azimuth $\{UEN\}$

Here:

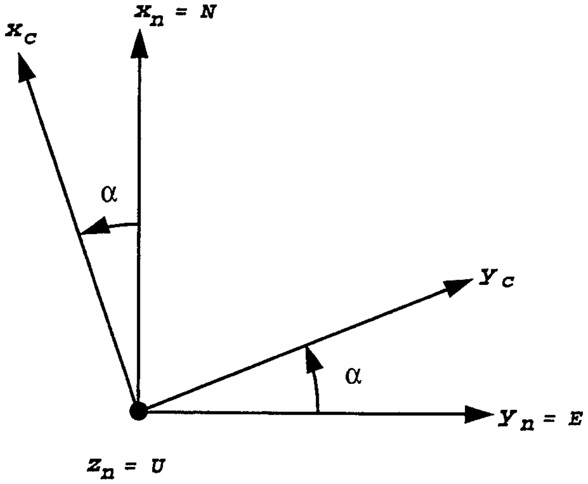


Figure 3.14: Wander Azimuth (NEU)

For $\{n\} = UEN$

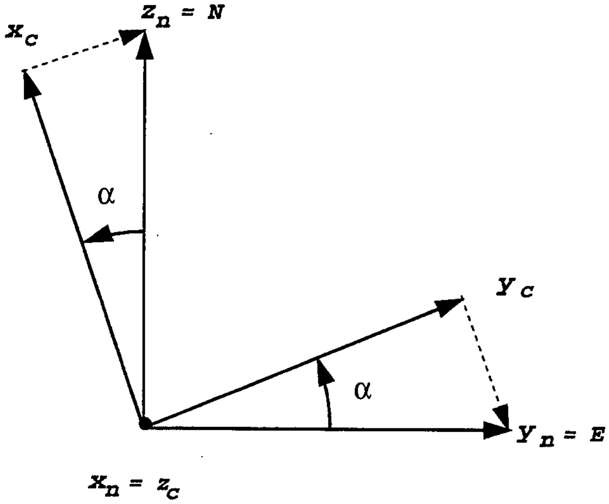


Figure 3.15: Wander Azimuth (UEN)

Look carefully at the orientation of $\{c\}$ and $\{n\}$. To understand each element of the transformation matrix, write out each component of $\{c\}$ separately and compare with the matrix form in Equation 3.34.

$$x_c = -y_n \sin\alpha + z_n \cos\alpha$$

$$
\begin{aligned}
y_c &= y_n \cos\alpha + z_n \sin\alpha \\
z_c &= x_n
\end{aligned}
$$

or

$$
\begin{bmatrix} x \\ y \\ z \end{bmatrix}_c = \begin{bmatrix} 0 & -s\alpha & c\alpha \\ 0 & c\alpha & s\alpha \\ 1 & 0 & 0 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \end{bmatrix}_n \tag{3.34}
$$

4. Navigation to Body: ${}^b_nC$

Define {b} & {n} using right hand rule ⇒ <u>NED</u>

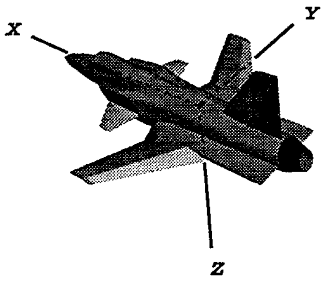


Figure 3.16: NED Orientation

Convention:

- ϕ positive when right wing down.
- θ positive when nose up.
- ψ positive when nose rotates N → E.

Then:

${}^b_nC = C_\phi C_\theta C_\psi,$

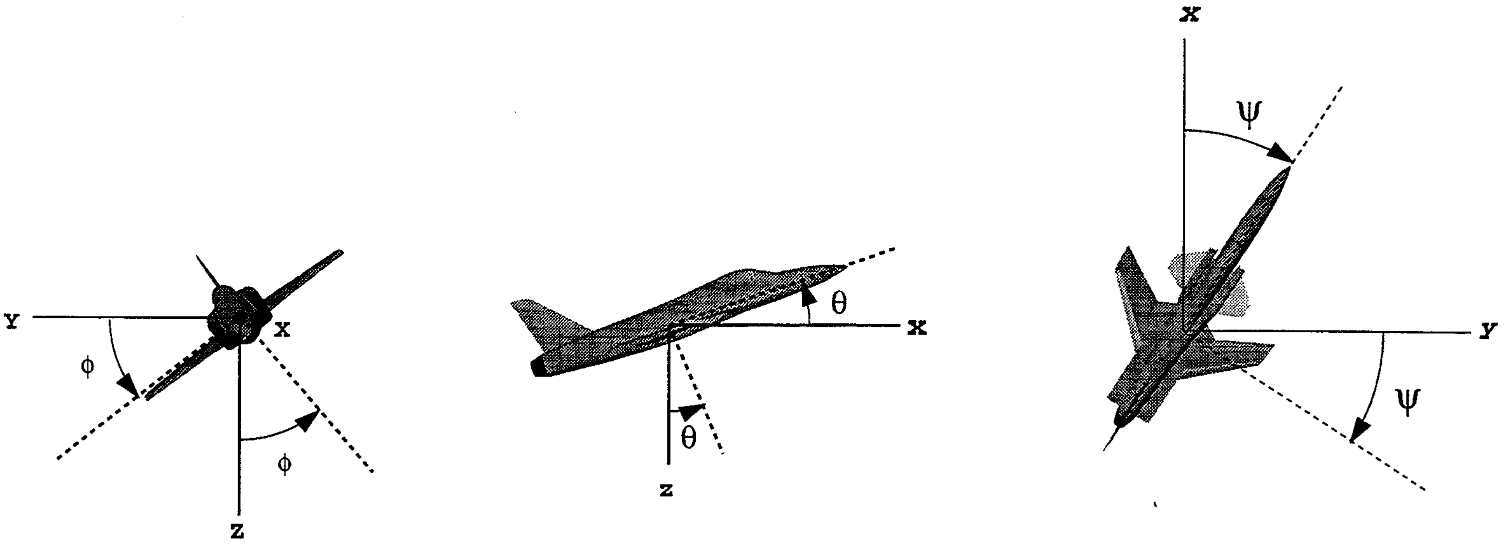


Figure 3.17: Positive ϕ, Positive θ, Positive ψ

where:

$$C_\phi = \begin{bmatrix} 1 & 0 & 0 \\ 0 & c\phi & s\phi \\ 0 & -s\phi & c\phi \end{bmatrix}, \; C_\theta = \begin{bmatrix} c\theta & 0 & -s\theta \\ 0 & 1 & 0 \\ s\theta & 0 & c\theta \end{bmatrix}, \; C_\psi = \begin{bmatrix} c\psi & s\psi & 0 \\ -s\psi & c\psi & 0 \\ 0 & 0 & 1 \end{bmatrix} \qquad (3.35)$$

I. COORDINATE FRAMES FOR INS MECHANIZATIONS

Types of INS configurations

- Local level (torqued)
- Space Stabilized
- Strap down

Local Level

The x, y accelerometers are always in a plane tangent to local ellipsoid (ideally $\{n\}$). Torque is applied to vertical axis to maintain certain orientation of x, y axes: There are four types of local–level mechanizations:

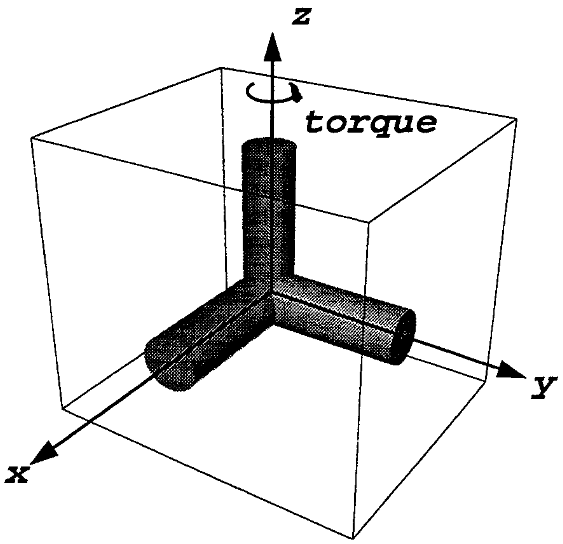


Figure 3.18: Torque Applied to Maintain Orientation

- North Slaved
- Unipolar
- Free Azimuth
- Wander Azimuth

In order to maintain a local–level orientation, the accelerometer platform must be torqued to align with the axes of the local–level frame. Each of these systems is defined by the azimuth torquing rate.

Question: How to compute the torque?

Answer: Must know angular velocity of platform with respect to the inertial frame.

Let: $\{n\} = ENU$

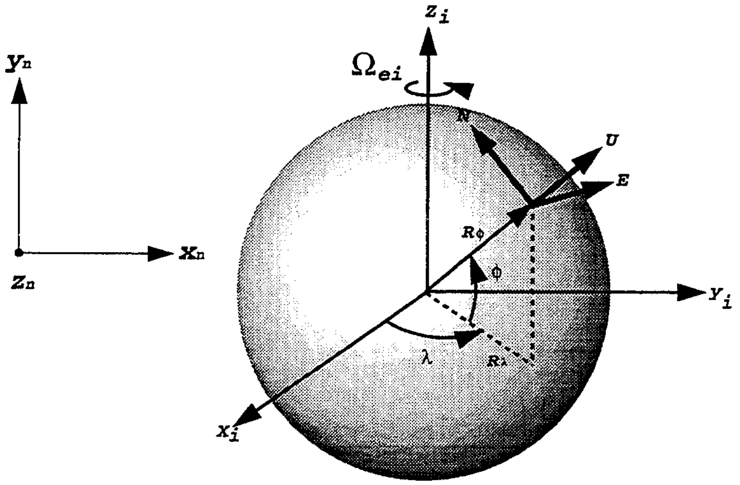


Figure 3.19: Geographic and Inertial Torque Model

Now,

$$\Omega_{ni} = \Omega_{ne} + \Omega_{ei}, \quad \text{where } \Omega_{ni} = \text{angular velocity of } \{n\} \ wrt \ \{i\} \tag{3.36}$$

Note: in Siouris [Ref. 1], ρ is used to define Ω_{ne}.

Recall Equation 3.26

$$\Omega_{ei} = \begin{bmatrix} 0 \\ \Omega \cos\phi \\ \Omega \sin\phi \end{bmatrix} \tag{3.37}$$

Let:

$$v = \begin{bmatrix} v_E \\ v_N \\ v_U \end{bmatrix} \tag{3.38}$$

be the velocity of $\{n\}$ wrt $\{e\}$.

Now, the angular velocity Ω_{ne} can be described as a function of the distance from the Earth's center and the linear velocity v at a point. Therefore, let:

$$\Omega_{ne} = \begin{bmatrix} \omega_E \\ \omega_U \\ \omega_N \end{bmatrix} \tag{3.39}$$

Consider Figure 3.20, the angular velocity ω_E can be expressed in terms of the radial distance and the linear velocity v_N, or:

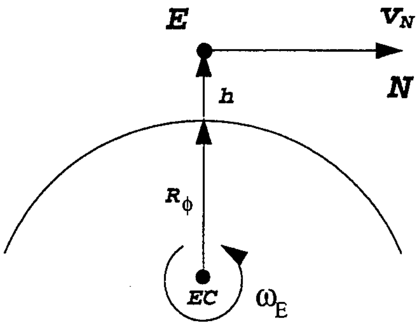


Figure 3.20: ω_E

$$\omega_E = -\frac{v_N}{R_\phi + h} \tag{3.40}$$

Next, from Figures 3.21 and 3.22 ω_N and ω_U are:

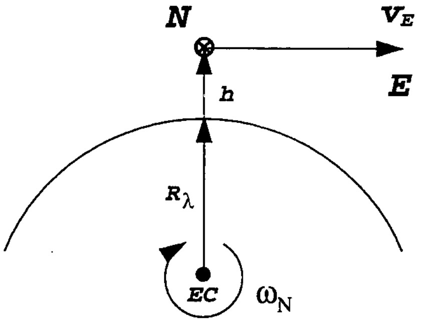


Figure 3.21: ω_N

$$\omega_N = \frac{v_E}{R_\lambda + h} \tag{3.41}$$

$$\omega_U = \omega_N \tan\phi = \frac{v_E}{R_\lambda + h}\tan\phi \tag{3.42}$$

For a spherical Earth, $R_\lambda = R_\phi = R_o$, otherwise,

$$R_\phi = \frac{R_e(1-\varepsilon^2)}{\sqrt{(1-\varepsilon^2\sin^2\phi)^3}}, \quad \text{where } R_e = \text{equatorial radius} \tag{3.43}$$

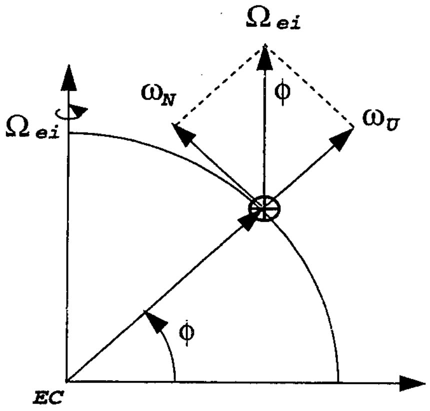


Figure 3.22: ω_U

$$R_\lambda = \frac{R_e}{\sqrt{1-\varepsilon^2 \sin^2\phi}} \tag{3.44}$$

where

$$\varepsilon = 1 - \frac{b^2}{a^2} = \text{ eccentricity factor}$$

and

$$a = \text{semi–major axis and } b = \text{semi–minor axis}$$

From Figure 3.23 we obtain

$$^{e}\Omega_{ni} = \begin{bmatrix} -\dot{\phi} \\ (\dot{\lambda}+\Omega)\cos\phi \\ (\dot{\lambda}+\Omega)\sin\phi \end{bmatrix} \tag{3.45}$$

Let the orientation of the navigation frame $\{n\}$ be UEN. Therefore, Ω_{ni} resolved in the $\{n\}$ frame is:

$$^{n}\Omega_{ni} = \begin{bmatrix} c\phi & 0 & s\phi \\ 0 & 1 & 0 \\ -s\phi & 0 & c\phi \end{bmatrix} \begin{bmatrix} c\lambda & s\lambda & 0 \\ -s\lambda & c\lambda & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 0 \\ 0 \\ \dot{\lambda}+\Omega \end{bmatrix} + \begin{bmatrix} c\phi & 0 & s\phi \\ 0 & 1 & 0 \\ -s\phi & 0 & c\phi \end{bmatrix} \begin{bmatrix} 0 \\ -\dot{\phi} \\ 0 \end{bmatrix} \tag{3.46}$$

$$^{n}\Omega_{ni} = \begin{bmatrix} (\dot{\lambda}+\Omega)s\phi \\ 0 \\ (\dot{\lambda}+\Omega)c\phi \end{bmatrix} + \begin{bmatrix} 0 \\ -\dot{\phi} \\ 0 \end{bmatrix} = \begin{bmatrix} (\dot{\lambda}+\Omega)s\phi \\ -\dot{\phi} \\ (\dot{\lambda}+\Omega)c\phi \end{bmatrix} \tag{3.47}$$

The negative sign in front of the $\dot{\phi}$ term comes from the orientation of the axes. Since the y-axis is east, the upward rotation of ϕ is negative using the right–hand–rule.

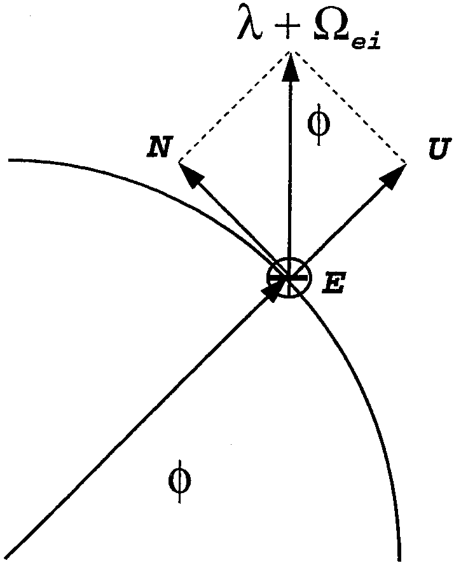


Figure 3.23: Ω_{ni}

To verify, point the thumb in the direction of the y-axis and curl the right hand. A positive rotation would be down instead of up. Hence, the negative sign.

Now convert to ENU orientation

$$^{n}\Omega_{ni} = \begin{bmatrix} -\dot{\phi} \\ (\dot{\lambda}+\Omega)c\phi \\ (\dot{\lambda}+\Omega)s\phi \end{bmatrix} \tag{3.48}$$

J. PLATFORM MECHANIZATIONS

Let α determine the drift (wander) angle of $\{c\} = \{p\}$ with respect to $\{n\} = \{g\}$, the geographic frame and is a function of both the sidereal rate, Ω_{ei} and Ω_{ne}, as shown in Equation 3.49.

$$\begin{bmatrix} 0 \\ 0 \\ \dot{\alpha} \end{bmatrix} = \begin{bmatrix} c\alpha & s\alpha & 0 \\ -s\alpha & c\alpha & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 0 \\ 0 \\ (\dot{\lambda}+\Omega)s\phi \end{bmatrix} \tag{3.49}$$

Figure 3.24 shows the commanded drift angle α. The rate of change of α or $\dot{\alpha}$ is the difference between the platform angular velocity about the z-axis ω_{zp} and the geographic angular velocity ω_{zg} or

$$\Rightarrow \dot{\alpha} = \omega_{zp} - \omega_{zg} = \omega_{zc} - \omega_{zg} \tag{3.50}$$

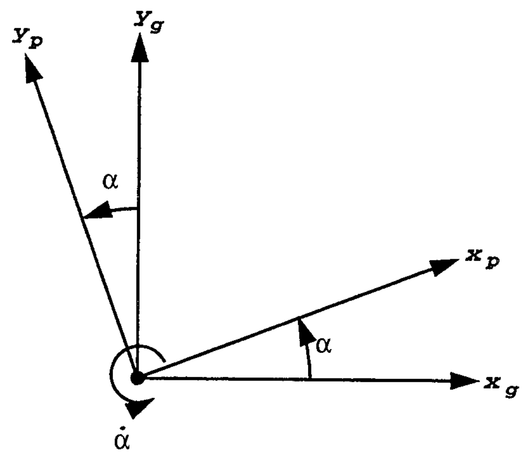


Figure 3.24: Platform Frame $\{p\}$ with Angular Rate, $\dot{\alpha}$

In order to keep the platform aligned we must command the angle α at a rate equal to ω_{zp}, we which we have defined as ω_{zc}. We can now define each mechanization in terms of $\dot{\alpha}$.

North Slaved

To maintain a north pointing system we must keep x_p pointing North, $\{NEU\}$ orientation. If we apply torque to keep $\dot{\alpha} = 0$

$$\Rightarrow \omega_{zc} = (\dot{\lambda} + \Omega)s\phi$$

Free–Azimuth

In a Free–Azimuth system, the vertical axis is not torqued, therefore:

$$\omega_{zc} = 0$$

Substituting into Equation 3.50, we get

$$\dot{\alpha} = -\omega_{zg} = -\Omega_{ei}\sin\phi$$

The equation at the top of page 46 in [Ref. 1] is incorrect and should be the same as Equation 3.50.

Unipolar

In the Unipolar system, the wander angle, α, is kept equal to the longitude, λ, of the geographic frame or $\dot{\alpha} = \pm\dot{\lambda}$. Substituting into Equation 3.50:

$$\omega_{zc} = \omega_{zg} \pm \dot{\lambda} \;\; \text{(sign changes when crossing the equator)}$$

Wander–Azimuth

In the Wander–Azimuth system, commanded torque ω_{zc} is equal to the vertical component of the earth's sidereal rate, $\Omega_{ei} \sin\phi$. Substituting into Equation 3.49:

$$\begin{aligned} \dot{\alpha} &= \Omega_{ei} \sin\phi - \Omega_{ei} \sin\phi - \dot{\lambda} \sin\phi \\ \dot{\alpha} &= \dot{\lambda} \sin\phi \\ & \quad \text{(No singularity at the pole)} \end{aligned}$$

Space Stabilized

Maintains constant orientation *wrt* inertial space.

Strap-down

Inertial sensors are mounted directly on the vehicle ⇒ transformation from sensor to inertial axis is computed rather than mechanized.

K. PLATFORM MISALIGNMENT

Platform to Accelerometer: ${}^{a}_{p}C$

Imperfections in accelerometer installation result in errors between $\{p\}$ and $\{a\}$. Assume misalignment angles are small: i.e. $s\phi \approx \phi$, $c\phi \approx 1$. Error angles:

- rotate by ϕ about x_p
- rotate by θ about y_p

- rotate by ψ about z_p

$$\Rightarrow \begin{bmatrix} c\psi & s\psi & 0 \\ -s\psi & c\psi & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} c\theta & 0 & -s\theta \\ 0 & 1 & 0 \\ s\theta & 0 & c\theta \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 \\ 0 & c\phi & s\phi \\ 0 & -s\phi & c\phi \end{bmatrix} \tag{3.51}$$

$$= \begin{bmatrix} 1 & \psi & 0 \\ -\psi & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & -\theta \\ 0 & 1 & 0 \\ \theta & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & \phi \\ 0 & -\phi & 1 \end{bmatrix} \tag{3.52}$$

$$= \begin{bmatrix} 1 & \psi & 0 \\ -\psi & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & \theta\phi & -\theta \\ 0 & 1 & \phi \\ \theta & -\phi & 1 \end{bmatrix} \tag{3.53}$$

$$= \begin{bmatrix} 1 & \theta\phi + \psi & -\theta + \psi\phi \\ -\psi & 1 & \phi + \psi\phi \\ \theta & -\phi & 1 \end{bmatrix} \tag{3.54}$$

Ignoring higher order terms, we get:

$${}^{a}_{p}C = \begin{bmatrix} 1 & \psi & -\theta \\ -\psi & 1 & \phi \\ \theta & -\phi & 1 \end{bmatrix} = I_{3x3} + \begin{bmatrix} 0 & \psi & -\theta \\ -\psi & 0 & \phi \\ \theta & -\phi & 0 \end{bmatrix} \tag{3.55}$$

$$= I_{3x3} + S(\Lambda), \Lambda = \begin{bmatrix} \phi \\ \theta \\ \psi \end{bmatrix} \tag{3.56}$$

$\Rightarrow$ ${}^{a}_{p}C$ is a linear function of Λ. In general, for small changes, Euler angles act like vectors. These small misalignment angles can be used to compensate for misalignment errors. (In this derivation we have assumed $\{a\}$ to be orthogonal).

L. ESTIMATE OF PLATFORM TO NAVIGATION TRANSFORMATION: ${}^{n}_{p}\hat{C}$

Let ${}^{n}_{p}\hat{C}$ be the estimate:

$${}^{n}_{p}\hat{C} = {}^{n}_{p}C + \delta\, {}^{n}_{p}C \text{ (small error)} \tag{3.57}$$

$$= \left(I + \delta\, {}^{n}_{p}C\, {}^{p}_{n}C\right) {}^{n}_{p}C \tag{3.58}$$

Since ${}^n_p\hat{C}\,{}^n_p\hat{C}^T = I$ we obtain:

$$\left(I + (\delta{}^n_pC)\,{}^p_nC\right)\,{}^n_pC\,{}^p_nC\left(I +{}^p_nC(\delta{}^n_pC)^T\right) = I$$

$$I + (\delta{}^n_pC)\,{}^n_pC + {}^p_nC(\delta{}^n_pC)^T + (\delta{}^n_pC)(\delta{}^n_pC)^T = I \tag{3.59}$$

Since $\delta C \delta C^T \approx 0$, we get:

$$\Rightarrow (\delta{}^n_pC){}^n_pC = -\left[(\delta{}^n_pC){}^n_pC\right]^T := M \tag{3.60}$$

Post multiplying the left and right sides of Equation 3.60 by ${}^p_nC$ gives:

$$\delta\,{}^n_pC = M\,{}^p_nC \tag{3.61}$$

Adding ${}^n_pC$ to both sides and remembering that ${}^p_nC = -{}^n_pC$ for a skew symmetric matrix, Equation 3.61 becomes:

$$\Rightarrow{}^n_p\hat{C} = (I - M){}^n_pC \tag{3.62}$$

IV. INERTIAL SENSORS

A. LASER GYRO FUNDAMENTALS: PASSIVE SAGNAC INTERFEROMETER

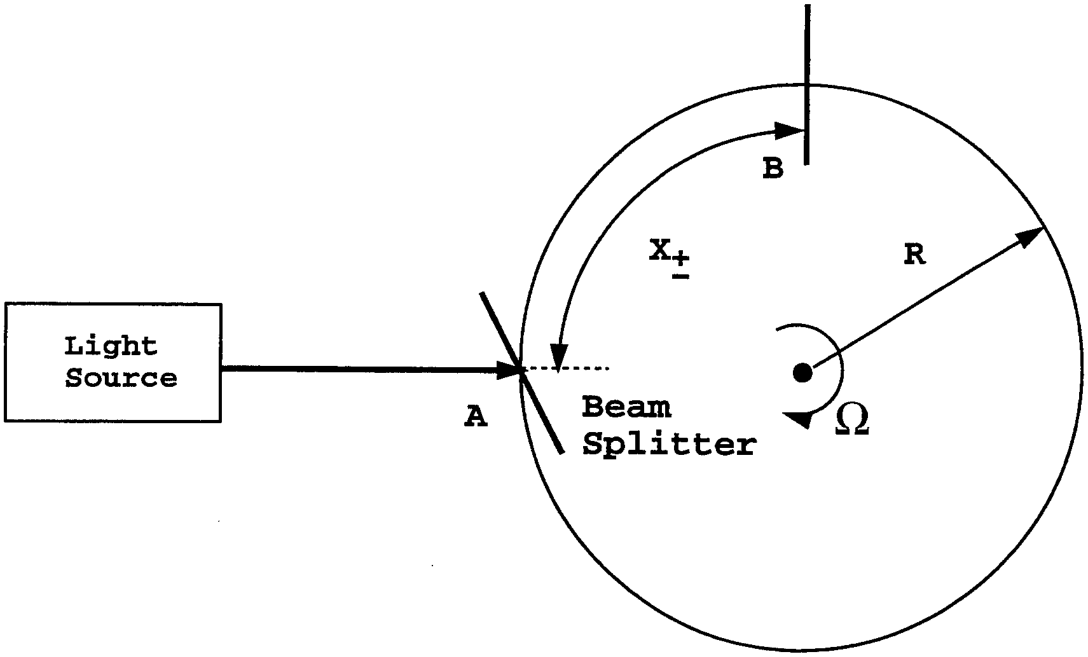


Figure 4.1: Ideal Rotating Sagnac Interferometer [Ref. 1, p. 87]

Light is split at A to travel cw and ccw. Suppose $\Omega = 0$. Then:

$$t = \frac{2\pi R}{c}$$

Suppose $\Omega \neq 0$. Let X be the distance A traveled in inertial space for light to return to beam-splitter. Then:

$$t_{\pm} = 2\pi R \pm X_{\pm}, \text{where} X_{\pm} = R\Omega t_{\pm}$$

Then:

$$t_{\pm} = \frac{2\pi R}{c} + \frac{R\Omega t_{\pm}}{c} \Rightarrow t_{\pm} = \frac{2\pi R}{c \mp R\Omega}$$

Then:

$$\begin{aligned}
\Delta t &= t_+ - t_- = \frac{2\pi R}{c - R\Omega} - \frac{2\pi R}{c + R\Omega} \\
&= 2\pi R \left[\frac{2R\Omega}{c^2 - R^2\Omega^2}\right] \\
&= \frac{4\pi R^2\Omega}{c^2 - R^2\Omega^2} \\
&= \left[\frac{\frac{4\pi R}{c}\frac{R\Omega}{c}}{1} - (\frac{R\Omega}{c})^2\right] \\
&= \frac{4\pi R}{c}\frac{R\Omega}{c}\left[1 + \frac{R\Omega}{c} + \left[\frac{R\Omega}{c}\right]^2 + \cdots\right]
\end{aligned}$$

Now, to a first approximation:

$$\Delta t = \frac{4\pi\Omega R^2}{c^2} \Leftarrow \text{ Sagnac effect}$$

The optical path difference:

$$\Delta L = c\Delta t = c\left[\frac{4\pi\Omega R^2}{c^2}\right] = \frac{4\pi R^2}{c} = \frac{4A\Omega}{c}$$

$$A = 4\pi R^2 \Leftarrow \text{ area enclosed by the circular path}$$

Sagnac effect is used in RLG and FOG.

B. FIBER OPTIC GYRO

Ω is measured by analyzing the phase shift caused by Sagnac effect

- Ideal behavior assumes reciprocity $\Rightarrow$ the paths of *cw* and *ccw* beams are identical $\Rightarrow$ the phase shift is due to inertial distance traveled

- Non-reciprocity may arise from non-linear index changes caused by unequal intensities $\Rightarrow$ is handled by using broadband source

- Non-reciprocity due to external magnetic fields like Faraday effect can be reduced by magnetic shielding

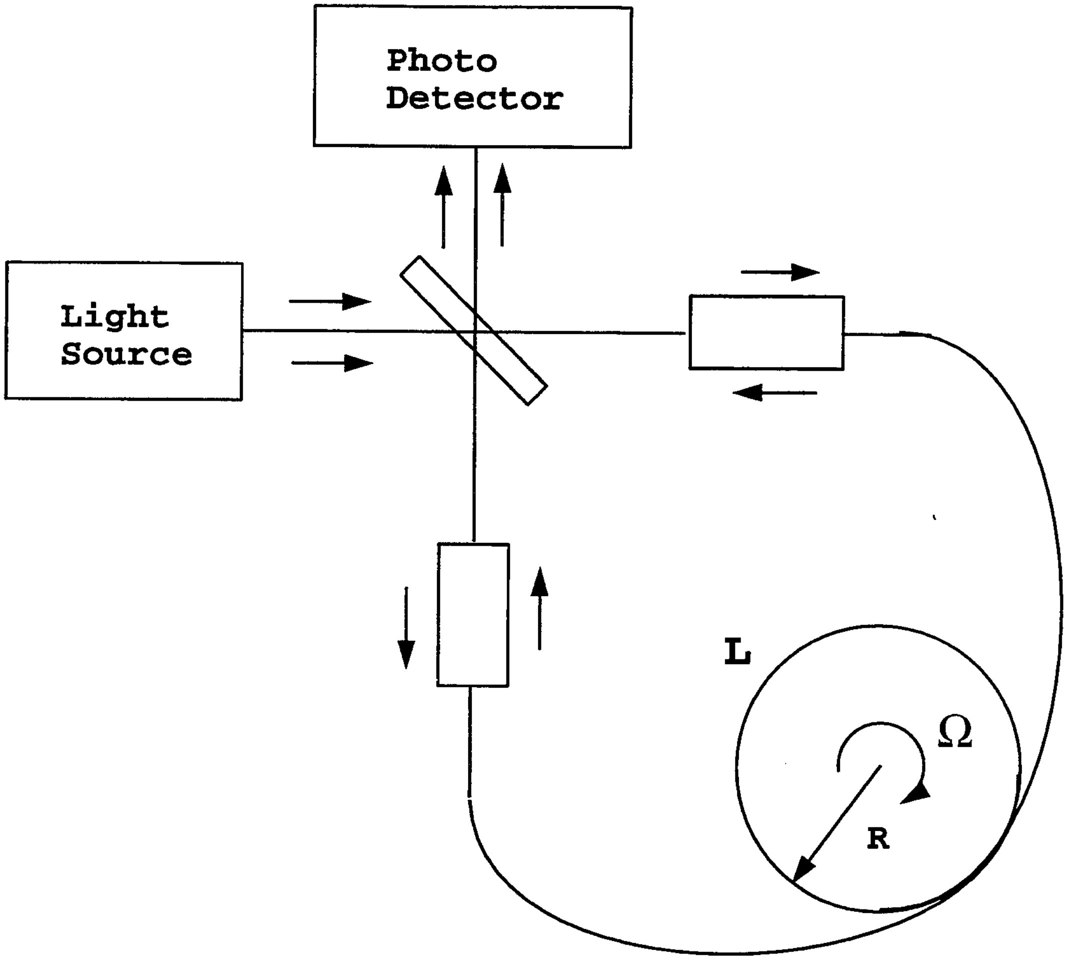


Figure 4.2: Fiber Optic Gyro [Ref. 1, p. 119]

- Fiber length: 50m to 1km

FOG elements:

- Laser diode that acts as a light source
- beam splitter
- coil of optical fiber
- photo detector

Light from laser diode is split into two beams ⇒ *cw* & *ccw*. The beams are superimposed and the resulting interference is monitored ⇒ measure phase shift by using photo detector ⇒ convert light into electrical current. Current is sampled every frame. Let λ be the wavelength of the light. Then:

$$\begin{aligned} T &= \frac{\lambda}{c}, \\ \text{and phase shift } \Delta\phi &= \frac{2\pi\Delta t}{T} \\ &= 2\pi\frac{4\pi\Omega R^2}{c^2}\frac{c}{\lambda} \\ &= \frac{[2\pi 2\pi(2\pi R)R\Omega]}{\lambda c} \\ &= \frac{4\pi LR}{\lambda c}\Omega, \text{ for } N = 1, \text{ number of turns} \end{aligned}$$

For $N > 1$, $L = 2\pi RN$ and $A = \pi R^2$

$$\begin{aligned} \Rightarrow \Delta\phi &= \frac{8\pi NA}{\lambda c}\Omega \\ K &= \frac{8\pi NA}{\lambda c} \text{ sensor scale factor} \end{aligned}$$

Note, Sagnac effect can be viewed in terms of Δt or $\Delta\phi$.

⇒ Two practical approaches to obtain Ω:

1. Measure $\Delta\phi$ ($\Rightarrow \Delta t$)

2. Change the frequency of one of the beams until $\Delta\phi = 0$.

Then:

$$\Delta f = \left(\frac{2R}{n\lambda}\right)\Omega \tag{4.1}$$

V. TANGENT PLANE NAVIGATION

A. TANGENT PLANE EQUATIONS

Consider the case of flight in a limited region. Use tangent plane:

$$X = \text{North}$$

$$Y = \text{East}$$

$$Z = \text{Up: Away from the earth's center}$$

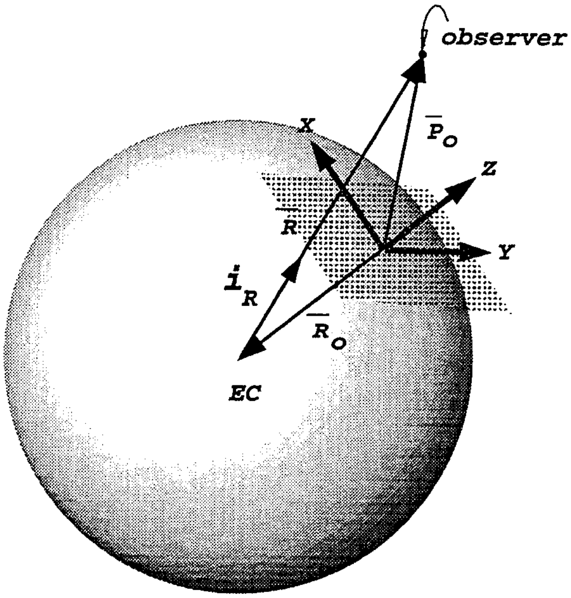


Figure 5.1: Tangent Plane

Let i_R be the unit vector in the direction away from the earth's center. Then, the local gravity vector g acting on the observer is

$$g = -\frac{GM}{R^2} i_R$$

where

$$\begin{aligned}\bar{R} &= \bar{P}_o - \bar{R}_o \text{ and } R = |\bar{R}|,\\ \bar{R}_o &= \text{vector connecting E.C. with the origin of the tangent plane.}\\ \bar{P}_o &= \text{position of the observer in the tangent plane.}\end{aligned}$$

Let $P_o = (x, y, z)$ then,

$$||R|| = \sqrt{x^2 + y^2 + (z + R_o)^2}$$

$$\bar{R} = \begin{bmatrix} x \\ y \\ z \end{bmatrix} - \begin{bmatrix} 0 \\ 0 \\ -R_o \end{bmatrix} = \begin{bmatrix} x \\ y \\ z + R_o \end{bmatrix}$$

$$i_R = \begin{bmatrix} \frac{x}{||R||} \\ \frac{y}{||R||} \\ \frac{z+R_o}{||R||} \end{bmatrix}$$

Therefore , the local gravity vector is:

$$g = \begin{bmatrix} -\frac{GM}{||R||^3}x \\ -\frac{GM}{||R||^3}y \\ -\frac{GM(z+R_o)}{||R||^3} \end{bmatrix}$$

Note, for flights near the Earth's surface, $R \simeq R_o$, then:

$$\begin{cases} g_x \simeq -\frac{g_o}{R_o}x \\ g_y \simeq -\frac{g_o}{R_o}y \\ g_z \simeq -g_o + 2\frac{g_o}{R_o}z \end{cases} \tag{5.1}$$

Where

$$g_o = \frac{GM}{R_o^2}$$

B. GENERAL NAVIGATION EQUATIONS

- Idea: Compute vehicle's position based on accelerometer readout & equations derived above.
- Constraint: Einstein's Uncertainty Principle ⇒ Accelerometers cannot distinguish between inertial and gravitational forces.

Therefore: Let

$$\begin{aligned} R &= \text{geocentric position vector of vehicle} \\ {}^{i}V &= \text{vehicle's inertial velocity} \end{aligned}$$

Then

$$\left\{ \begin{array}{lcl} \dot{R} &=& {}^{i}V \\ {}^{i}\dot{V} &=& A + g_m(R) \Leftarrow \text{ Einstein's principle} \end{array} \right. \tag{5.2}$$

Where

$$\begin{aligned} A &= \text{specific force measured by accelerations} \\ g_m(R) &= \text{gravitational acceleration} \end{aligned}$$

Equation 5.2 represents a system of differential equations that can be integrated to get vehicle's position in an inertial frame.

Recall, $\ddot{r} = A + g$, and we get:

$$\left\{ \begin{array}{lcl} A_x &=& \ddot{x} - g_x = \ddot{x} + \frac{g_o}{R_o}x \\ A_y &=& \ddot{y} - g_y = \ddot{y} + \frac{g_o}{R_o}y \\ A_z &=& \ddot{z} - g_z \approx \ddot{z} - 2\frac{g_o}{R_o}z + g_o \end{array} \right. , \tag{5.3}$$

where

$$-\left[g_o + \left. \frac{GMR^3 - GM(z+R_O)3R^2\frac{2(z+R_o)}{2R}}{R^6} \right|_{z=0} z \right]$$

$$\begin{aligned} -\frac{gM(z+R_o)}{R^3} &= -\left(g_o + \frac{\partial}{\partial z}\left[\frac{gM(z+R_o)}{R^3} \right] \right) \\ &= -\left(g_o + \left. \frac{\left[GMR^3 - (GM(z+R_O)3R^2)\left(\frac{2(z+R_o)}{2R} \right) \right]}{R^6} \right|_{z=o} z \right) \\ &= -\left(g_o - \frac{2GMRR_o}{R^6} z \right) \\ &\approx -g_o + \frac{2g_o}{R_o} z \end{aligned}$$

The mechanization for A_x, A_y, A_z is shown in Figure 5.2.

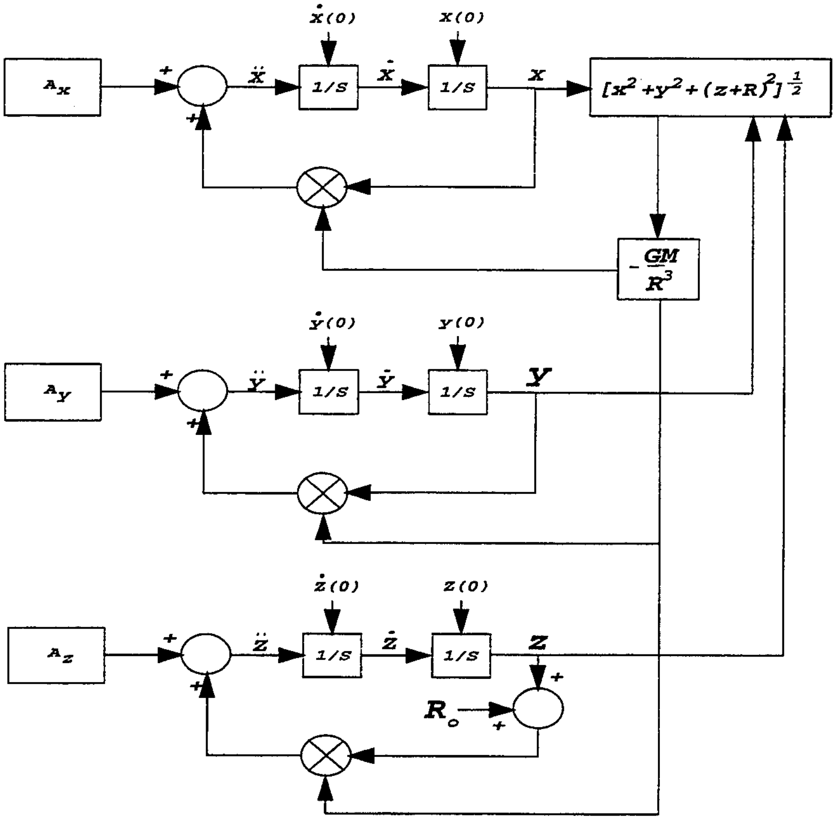


Figure 5.2: Block Diagram of Inertial Navigation System

Solve the differential equations in Equation 5.3 for $\frac{x}{R_o}$, $\frac{y}{R_o}$, and $\frac{z}{R_o}$ to get:

$$\begin{aligned}
\frac{x}{R_o} &= \frac{A_x}{g_o}\left(1-\cos\sqrt{\frac{g_o}{R_o}}t\right)\\
\frac{y}{R_o} &= \frac{A_y}{g_o}\left(1-\cos\sqrt{\frac{g_o}{R_o}}t\right)\\
\frac{z}{R_o} &= \frac{A_z-g_o}{2g_o}\left[\cosh\sqrt{\frac{2g_o}{R_o}}t-1\right]
\end{aligned}$$

Therefore, for constant thrust acceleration in x and y direction the vehicle is displaced to an average position such that the gravitational component in x or y direction is equal to imposed thrust acceleration. The vehicle oscillates about this mean position with a period of:

$$T = \frac{1}{2\pi}\sqrt{\frac{R_o}{g_o}} \simeq 84 \text{ min}$$

Thrust acceleration in the z-direction greater than g_o causes exponentially increasing velocity and position.

C. ERROR ANALYSIS

Linearize eq. 5.3 at (x_o, y_o, z_o);

$$\left\{ \begin{array}{lcl} \delta\ddot{x} & = & -\frac{g_o}{R_o}\delta x + \delta Ax \\ \delta\ddot{y} & = & -\frac{g_o}{R_o}\delta y + \delta Ay \\ \delta\ddot{z} & = & 2\frac{g_o}{R_o}\delta z + \delta Az \end{array} \right. \tag{5.4}$$

Therefore:

$$\begin{aligned} \frac{\delta x}{R_o} &= \frac{\delta Ax}{g_o}\left(1 - \cos\sqrt{\frac{g_o}{R_o}t}\right) = \frac{\delta y}{R_o} \\ \frac{\delta z}{R_o} &= \frac{\delta Az}{2g_o}\left(\cosh\sqrt{\frac{2g_o}{R_o}t - 1}\right) \end{aligned}$$

Therefore, errors in x and y are sinusoidal, but errors in z increase exponentially with time. ⇒ do not use dead reckoning to compute z.

D. THE VERTICAL CHANNEL

- Integrating acceleration in z leads to divergent solution
- Use altimeter measurements together with a_z to obtain altimeter data.
- Altitude is computed in CADC (Central Air Data Computer) based on
  - T_{FAT} = free air temperature
  - P_s =static pressure

Definition's

- Absolute altitude (H_{abs}) Height above the earth surface at a given location.

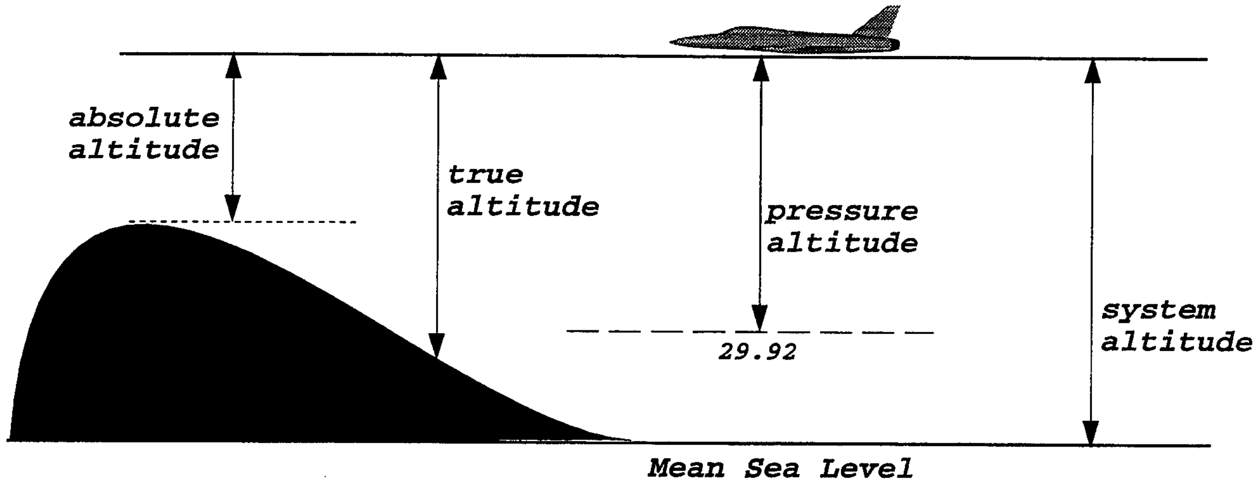


Figure 5.3: Altitude Definitions [Ref. 1, p. 199]

- True altitude (H_{true}) Actual height above the standard sea level.

- Pressure altitude (H_p) Height in model atmosphere above pressure datum plane of 29.92 in. (760mm) of mercury. (Does not consider variations in pressure and temperature at sea level, unlike true h).

- System altitude Computer corrected H_p for non-standard day variations.

ICAO Standard Atmosphere [Ref. 1, p. 201].

- Earths atmosphere
  - Troposphere: Lowest layer. Characterized by decrease in atmosphere (negative lapse rate). Contains most of air and moisture.
  - Stratosphere: Lapse rate changes direction. Marked decrease in water vapor.
  - Metosphere: Lapse rate again reverse and T decreases with altitude.

- Thermosphere: Above 80km, air temperature decreases with altitude. Gas molecules are separated by large distances.

E. ATMOSPHERIC MODEL

$$P = \text{pressure}$$

$$Z = \text{geometric altitude}$$

$$dP = -\rho g dz \tag{5.5}$$

Where

$$g = \frac{g_o R_e^2}{(R_e + z)^2}$$

$$g_o = 9.87m/s^2(32.2ft/s^2)$$

$$\rho = \text{atmospheric density}$$

From the ideal gas law

$$\rho = \frac{MP}{R^*T} \tag{5.6}$$

Where

$$M = \text{mean molecular weight of the air}$$

$$R^* = \text{universal gas constant}$$

$$T = \text{absolute temperature}$$

$$\Rightarrow \frac{dP}{P} = -\frac{gM}{R^*T}dz$$

$$\Rightarrow d\ln P = -\frac{gM}{R^*T}dz$$

$$\Rightarrow P = P_o \exp -\frac{gM}{R^*T}z$$

Where P_o = pressure at sea level

Similarly

$$\rho = \rho_o \exp - \frac{gM}{R^*T} z$$

Let

$$\Gamma = -\frac{\partial T}{\partial z} \text{ (lapse rate)}$$

Then for a given layer

$$\begin{aligned}
T &= T_o - \Gamma z \\
\Gamma &= -\frac{\partial T}{\partial z} \\
P &= P_o \left(\frac{T}{T_o}\right)^{\left(\frac{gM}{R^*T}\right)-1} \\
\rho &= \rho_o \left(\frac{T}{T_o}\right)^{\left(\frac{gM}{R^*T}\right)-1}
\end{aligned}$$

Also, from US atmosphere model, we get:

$$T = \Gamma(H_p - H_o) + T_o$$

and from Equations 5.5 and 5.6

$$\begin{aligned}
H_p &= H_o + \frac{T_o\left(\frac{P}{P_o} - \frac{R^*\Gamma}{g_oM} - 1\right)}{\Gamma}, \Gamma \neq 0 \\
H_p &= H_o - \frac{R^*T_o}{g_oM} \ln\left(PP_o\right), \Gamma = 0
\end{aligned}$$

Where, T_o, H_o, P_o are obtained from US Standard Atmosphere Model. Numerical examples [Ref. 1, p. 204].

F. VERTICAL CHANNEL DAMPING: COMPLEMENTARY FILTERING

Idea: Use altimeter information in steady state and at low frequencies (altitude is noisy). Use acceleration data at higher frequencies.

How? Consider:

$$\begin{aligned} h &= \frac{s^2+as+b}{s^2+as+b}\,h \\ &= \frac{s^2}{s^2+as+b}h + \frac{as+b}{s^2+as+b}\,h \\ &= \frac{1}{s^2+as+b}\,s^2\,h + \frac{as+b}{s^2+as+b}\,h \end{aligned}$$

Note:

$$s^2h = \ddot{h} = \text{ vertical acceleration in inertial } z\text{-direction}$$

$\Rightarrow$ use ${}^{i}(a_z + g_z) \Rightarrow$ z-acceleration computation in inertial frame. The mechanization is shown in Figure 5.4.

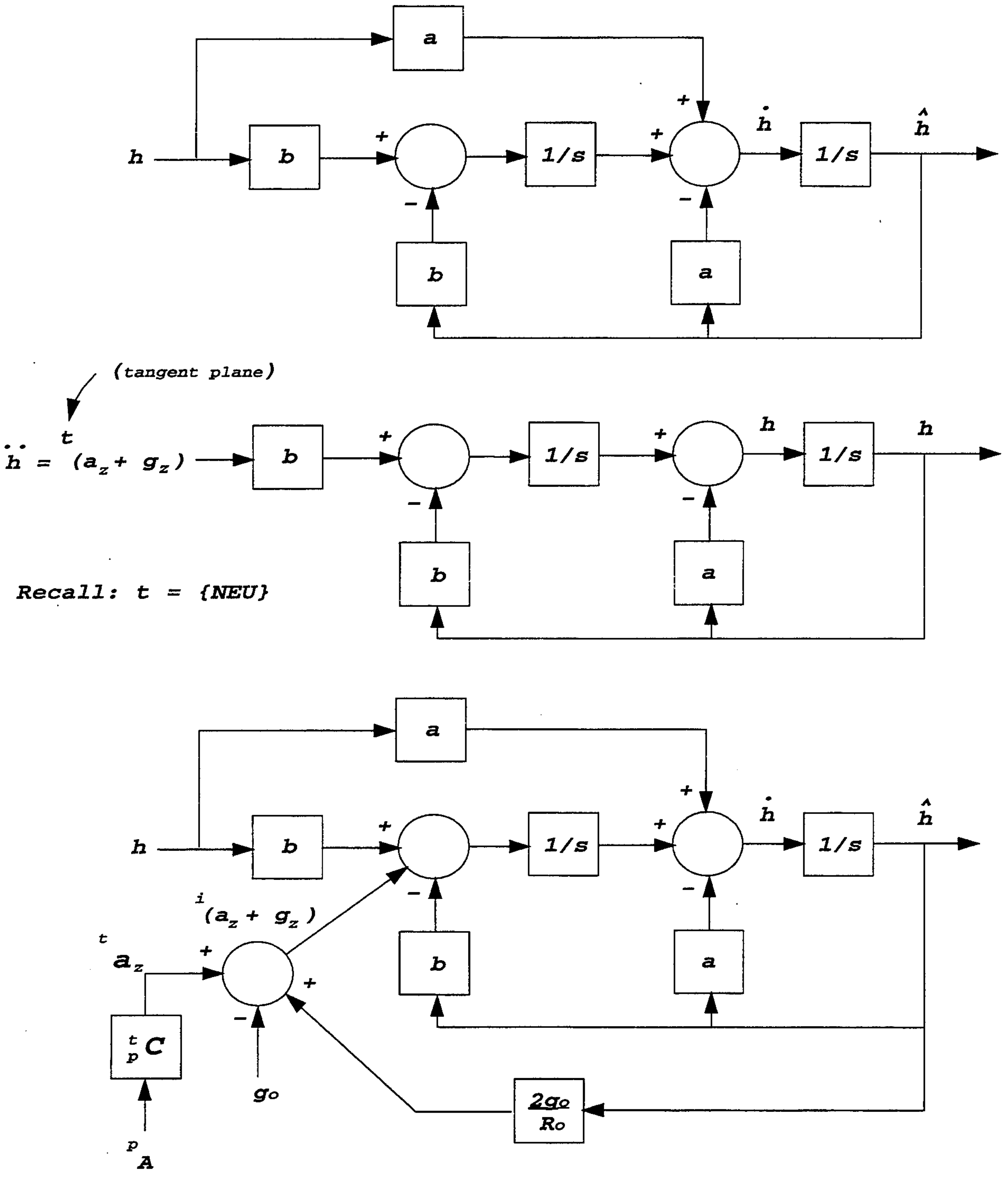


Figure 5.4: Complementary Filter Mechanization

Consider Figure 5.5 (see Figure 4-16 [Ref. 1]). We can write the following:

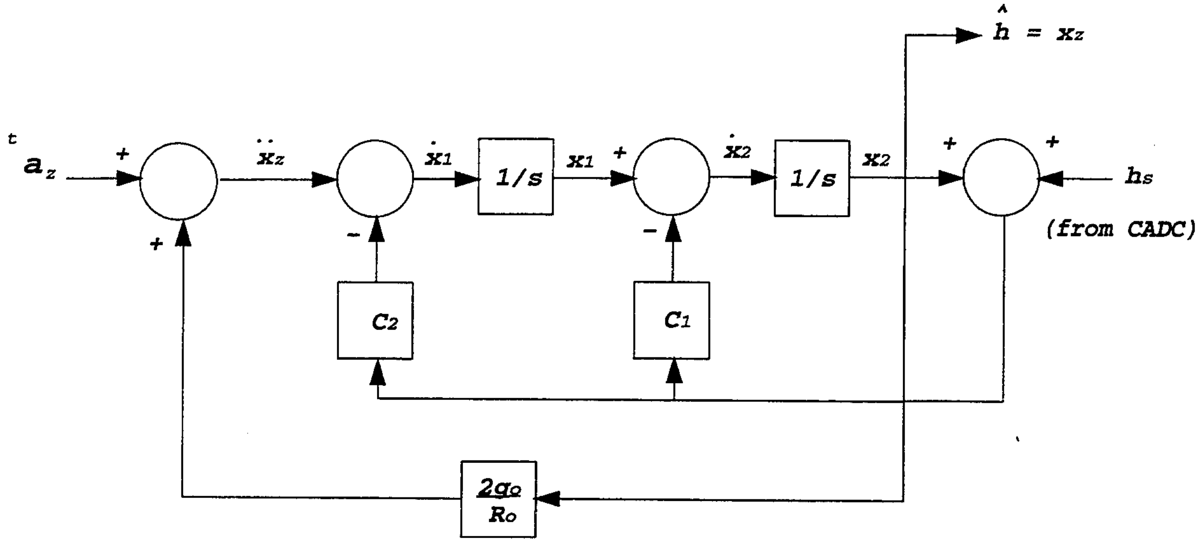


Figure 5.5: Complementary Filter: z–Channel

$$\left\{\begin{array}{lcl}\dot{x}_1 &=& \ddot{x}_z - c_2(h_s+x_2)\\ \dot{x}_2 &=& x_1 - c_1(h_s+x_2)\end{array}\right. \tag{5.7}$$

$$\Rightarrow\left\{\begin{array}{lcl}\dot{x}_1 &=& -c_2x_2 - c_2h_s + \ddot{x}_z\\ \dot{x}_2 &=& -c_1x_2 + x_1 - c_1h_s\end{array}\right.$$

In state space form we have:

$$\begin{bmatrix}\dot{x}_1\\ \dot{x}_2\end{bmatrix} = \underbrace{\begin{bmatrix}0 & -c_2\\ 1 & -c_1\end{bmatrix}}_{A}\begin{bmatrix}x_1\\ x_2\end{bmatrix} + \underbrace{\begin{bmatrix}-c_2\\ -c_1\end{bmatrix} h_s + \begin{bmatrix}1\\ 0\end{bmatrix}\ddot{x}_z}_{B}$$

$$\hat{h} = x_z = \underbrace{\begin{bmatrix}0 & 1\end{bmatrix}}_{C}\begin{bmatrix}x_1\\ x_2\end{bmatrix}$$

In transfer function form, $\hat{h}(s) = C(sI-A)^{-1}B$ we get:

$$\hat{h}(s) = \begin{bmatrix}0 & 1\end{bmatrix}\begin{bmatrix}s & c_2\\ -1 & s+c_1\end{bmatrix}^{-1}\left(\begin{bmatrix}-c_2\\ -c_1\end{bmatrix} h_s + \begin{bmatrix}1\\ 0\end{bmatrix}\ddot{x}_z\right) \tag{5.8}$$

Taking the inverse of $(SI-A)$:

$$(sI-A)^{-1} = \begin{bmatrix}0 & 1\end{bmatrix}\begin{bmatrix}s & c_2\\ -1 & s+c_1\end{bmatrix}^{-1} = \frac{1}{s^2+c_1s+c_2}\begin{bmatrix}s+c_1 & c_2\\ 1 & s\end{bmatrix} \tag{5.9}$$

Substituting 5.9 into 5.8 and multiplying through gives:

$$\begin{aligned}\hat{h}(s) &= \frac{1}{s^2+c_1s+c_2}\begin{bmatrix} 0 & 1 \end{bmatrix}\left(\begin{bmatrix} -(s+c_1)c_2-c_1c_2 \\ -c_2-sc_1 \end{bmatrix} h_s + \begin{bmatrix} s+c_1 \\ 1 \end{bmatrix}\ddot{x}_z\right) \\ &= \frac{-(sc_1+c_2)}{s^2+c_1s+c_2}\,h_s + \frac{1}{s^2+c_1s+c_2}\,\ddot{x}_z\end{aligned}$$

Compare with our filter:

$$c_1 = a,\;\; c_2 = b$$

Now add a_z:

$$\left\{\begin{array}{lcl} \dot{x}_1 &=& a_z + 2\omega_s^2 x_2 - c_2(h_s+x_2) \\ \dot{x}_2 &=& x_1 - c_1(h_s+x_2) \end{array}\right. \tag{5.10}$$

$$\Rightarrow\left\{\begin{array}{lcl} \dot{x}_1 &=& (-c_2+2\omega_s^2)x_2 - c_2h_s + a_z \\ \dot{x}_2 &=& -c_1x_2 + x_1 - c_1h_s \end{array}\right. \tag{5.11}$$

$$\begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \end{bmatrix} = \begin{bmatrix} 0 & -\overbrace{c_2+2\omega_s^2}^{\hat{c}_2} \\ 1 & -c_1 \end{bmatrix}\begin{bmatrix} x_1 \\ x_2 \end{bmatrix} + \begin{bmatrix} -c_2 \\ -c_1 \end{bmatrix} h_s + \begin{bmatrix} 1 \\ 0 \end{bmatrix}\ddot{x}_z$$

$$\hat{h} = x_z = \begin{bmatrix} 0 & 1 \end{bmatrix}\begin{bmatrix} x_1 \\ x_2 \end{bmatrix}$$

This gives the transfer function $\hat{h}(s)$:

$$\Rightarrow \hat{h}(s) = \frac{-(sc_1+\hat{c}_2)}{s^2+c_1s+\hat{c}_2}\,h_s + \frac{1}{s^2+c_1s+\hat{c}_2}\,a_z$$

Where:

$$\Rightarrow\left\{\begin{array}{lcl} a &=& c_1 \\ \hat{b} &=& c_2 - 2\omega_s^2 = b - 2\omega_s^2 \end{array}\right.$$

Substituting:

$$\hat{h}(s) = \frac{as+b-2\omega_s^2}{s^2+as+b-2\omega_s^2}\,h_s + \frac{1}{s^2+as+b-2\omega_s^2}\,(a_z-g_o)$$

Note, in steady state a_z must equal g_o. Suppose a_z measurement has errors, i.e. let

$$a_z = a_{zo} + \delta a_z, \;\; a_{zo} = g_o$$

and suppose $\delta a_{zo} \neq 0$ in steady state. Compute $\hat{h}$ in steady state.

Using the Final Value Theorem:

$$\begin{aligned}
\hat{h}(0) &= \lim_{s\to 0} s\,\hat{h}(s) \\
&= \lim_{s\to 0} s\,T_1(s)\,h_s(s) + \lim_{s\to 0} s\,T_2(s)\delta a_z(s)
\end{aligned}$$

Where:

$$\begin{aligned}
T_1(s) &= \frac{as + b - 2\omega_s^2}{s^2 + as + b - 2\omega_s^2} \\
T_2(s) &= \frac{1}{s^2 + as + b - 2\omega_s^2}
\end{aligned}$$

Suppose:

$$\begin{aligned}
\delta a_z &= \text{const}_2 \Rightarrow \delta a_z(s) = \frac{\text{const}_2}{s} \\
h_s &= \text{const}_1 \Rightarrow h_s(s) = \frac{\text{const}_1}{s}
\end{aligned}$$

$$\begin{aligned}
\Rightarrow \hat{h}(0) &= T_1(0)h_s(0) + T_2(0)\delta a_z(0) \\
&= h_s(0) + \frac{1}{b - 2\omega_s^2}\delta a_z(0)
\end{aligned}$$

Note, we want $\hat{h}(0) = h_s(0)$.

Consider

$$\begin{aligned}
\hat{h}(s) &= \frac{s^3 + as^2 + bs + \hat{c}}{s^3 + as^2 + bs + \hat{c}} h_s \\
&= \underbrace{\frac{s^3 + bs + \hat{c}}{s^3 + as^2 + bs + \hat{c}}}_{\hat{T}_1(s)} h_s(s) + \underbrace{\frac{as^2}{s^3 + as^2 + bs + \hat{c}}}_{\hat{T}_2(s)} \delta a_z(s)
\end{aligned}$$

Where

$$\hat{c} = c - 2\omega_s^2 = c - \frac{2g_o}{R_o}$$

Now, let $h_s(0) = \text{const}_1$, $\delta a_z(0) = \text{const}_2$

$$\hat{h}(0) = \lim_{s \to 0} \left[s\, \hat{T}_1(s)\, \hat{h}(s) + s\, \hat{T}_2(s)\, \delta a_z(s) \right]$$

Note,

$$\begin{aligned} s\, \hat{T}_1(s)\, h_s(s)|_{s=0} &= \hat{T}_1(0) = 1 \\ s\, \hat{T}_2(s)\, \delta a_z(s)|_{s=0} &= s\, \hat{T}_2(s) \frac{\text{const}_2}{s} = \hat{T}_2(0) = 0 \end{aligned}$$

$$\Rightarrow \hat{h}(0) = h_s(0)$$

Now the bandwidth of $\hat{T}_1(s)$ is selected based on the bandwidth of the altitude sensor.

VI. KINEMATIC EQUATIONS AND ERROR ANALYSIS

A. ROTATING COORDINATES

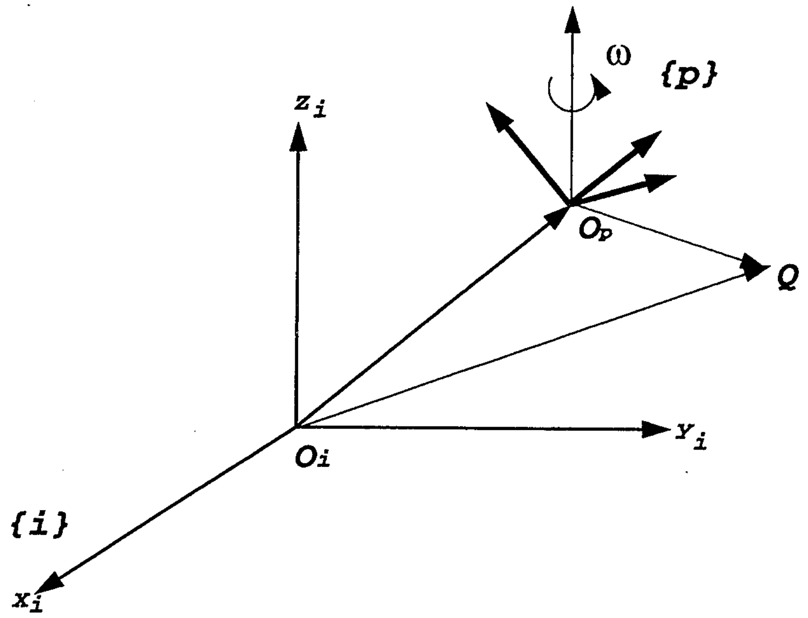


Figure 6.1: Rotating Coordinates

$$ {}^{i}P_Q = {}^{i}P_{O_p} + {}^{i}_{p}C\,{}^{p}P_Q \tag{6.1}$$

Suppose:

$$ {}^{i}P_{O_p} = 0 \text{ and } {}^{p}P_Q = \text{const} $$

Then:

$$ {}^{i}P_Q = {}^{i}_{p}C\,{}^{p}P_Q $$

Differentiating:

$$ \frac{d}{dt}\,{}^{i}P_Q = \frac{d}{dt}\,{}^{i}_{p}C\,{}^{p}P_Q = \frac{d}{dt}({}^{i}_{p}C){}^{p}P_Q $$

Note

$$\begin{aligned}
\frac{d}{dt}\,{}^i_pC &= {}^i_pC\,S({}^p\Omega_{pi}) \\
&= {}^i_pC\,S({}^p_iC\,{}^i\Omega_{pi}) \\
&= {}^i_pC\,{}^p_iC\,S({}^i\Omega_{pi})\,{}^p_iC^T \\
&= S({}^i\Omega_{pi})\,{}^i_pC
\end{aligned}$$

$$\begin{aligned}
\Rightarrow \frac{d}{dt}\,{}^iP_Q &= S({}^i\Omega_{pi})\,{}^i_pC\,{}^pP_Q \\
&= {}^i\Omega_{pi} \times {}^iP_Q \\
&= {}^i\omega \times {}^iP_Q
\end{aligned}$$

Now, suppose ${}^pP_Q \neq$ const. Let $\frac{\delta}{\delta t}\,{}^iP_Q =$ vector of time derivatives of each element of ${}^iP_Q$.

Then:

$$\frac{d}{dt}\,{}^iP_Q = \frac{\delta}{\delta t}\,{}^iP_Q + {}^i\omega \times {}^iP_Q$$

In general, Coriolis theorem:

$$\frac{d}{dt}A = \frac{\delta}{\delta t}A + \omega \times A \tag{6.2}$$

Now, let's compute:

$$\begin{aligned}
\frac{d^2}{dt^2}\,{}^iP_Q &= \frac{d}{dt}\left(\frac{\delta}{\delta t}\,{}^iP_Q\right) + \frac{d}{dt}\left({}^i\omega \times {}^iP_Q\right) \\
&= \frac{\delta^2}{\delta t^2}\,{}^iP_Q + {}^i\omega \times \frac{\delta}{\delta t}\,{}^iP_Q + \frac{\delta}{\delta t}\left({}^i\omega \times {}^iP_Q\right) + {}^i\omega \times \left({}^i\omega \times {}^iP_Q\right)
\end{aligned}$$

Note:

$$\frac{d}{dt}(a \times b) = \dot{a} \times b + a \times \dot{b} \tag{6.3}$$

Therefore:

$$\frac{\delta}{\delta t}\left({}^{i}\omega\right) \times {}^{i}P_Q = \left(\frac{\delta}{\delta t}{}^{i}\omega\right) \times {}^{i}P_Q + {}^{i}\omega \times \frac{\delta}{\delta t}{}^{i}P_Q \tag{6.4}$$

Note

$$\frac{d}{dt}{}^{i}\omega = \frac{\delta}{\delta t}{}^{i}\omega + \overbrace{{}^{i}\omega \times {}^{i}\omega}^{0} = \frac{\delta}{\delta t}{}^{i}\omega \tag{6.5}$$

Therefore, we get:

$$\begin{aligned} \frac{d^2}{dt^2}{}^{i}P_Q &= \frac{\delta^2}{\delta t^2}{}^{i}P_Q + {}^{i}\omega \times \frac{\delta}{\delta t}{}^{i}P_Q + \left(\frac{d}{dt}{}^{i}\omega\right) \times {}^{i}P_Q + \\ &\quad {}^{i}\omega \times \frac{\delta}{\delta t}{}^{i}P_Q + {}^{i}\omega \times \left({}^{i}\omega \times {}^{i}P_Q\right) \\ &= \frac{\delta^2}{\delta t^2}{}^{i}P_Q + \left(\frac{d}{dt}{}^{i}\omega\right) \times {}^{i}P_Q + 2\,{}^{i}\omega \times \frac{\delta}{\delta t}{}^{i}P_Q + {}^{i}\omega \times \left({}^{i}\omega \times {}^{i}P_Q\right) \end{aligned}$$

Notation $\frac{d}{dt}x = \dot{x}$

Now, suppose ${}^{i}P_{O_p} \neq 0$, then

$${}^{i}\ddot{P}_Q = {}^{i}\ddot{P}_{O_p} + \frac{\delta^2}{\delta t^2}{}^{i}P_Q + {}^{i}\dot{\Omega}_{pi} \times {}^{i}P_Q + 2\,{}^{i}\Omega_{pi} \times \frac{\delta}{\delta t}{}^{i}P_Q + {}^{i}\Omega_{pi} \times \left({}^{i}\Omega_{pi} \times {}^{i}P_Q\right)$$

Using the notation in Siouris [Ref. 1],

$$\begin{aligned} {}^{i}P_Q &= p \\ P_{O_p} &= R \\ r &= R + p \end{aligned}$$

Then:

$$\ddot{r} = \ddot{R} + \frac{\delta p^2}{\delta t^2} + {}^{i}\dot{\Omega}_{pi} \times p + {}^{i}\Omega_{pi} \times ({}^{i}\Omega_{pi} \times p) + 2\,{}^{i}\Omega_{pi} \times \frac{\delta p}{\delta t} \tag{6.6}$$

Where:

$$\begin{aligned}
\ddot{r},\ \ddot{R},\ \frac{\delta^2 p}{\delta t^2} &= \text{linear acceleration terms} \\
{}^i\dot{\Omega}_{pi} \times p &= \text{tangential component of acceleration due to } {}^i\dot{\Omega}_{pi} \\
{}^i\Omega_{pi} \times ({}^i\Omega_{pi} \times p) &= \text{centripetal acceleration} \\
2\,{}^i\Omega_{pi} \times \frac{\delta p}{\delta t} &= \text{Coriolis acceleration}
\end{aligned}$$

Recall the general navigation equations.

- Idea: Compute vehicle's position based on accelerometer readout & equations derived above.
- Constraint: Einstein's Uncertainty Principle $\Rightarrow$ Accelerometers cannot distinguish between inertial and gravitational forces.

Therefore: Let

$$\begin{aligned}
R &= \text{geocentric position vector of vehicle} \\
{}^iV &= \text{vehicle's inertial velocity}
\end{aligned}$$

Then

$$\left\{ \begin{aligned} \dot{R} &= {}^iV \\ {}^i\dot{V} &= A + g_m(R) \Leftarrow \text{Einstein's principle} \end{aligned} \right. \tag{6.7}$$

Where

$$\begin{aligned}
A &= \text{specific force measured by accelerations} \\
g_m(R) &= \text{gravitational acceleration}
\end{aligned}$$

Equation 6.7 represents a system of differential equations that can be integrated to get vehicle's position in an inertial frame. As shown in Figure 6.2 the accelerometers measure ${}^pA$ = specific force resolved in $\{p\}$.

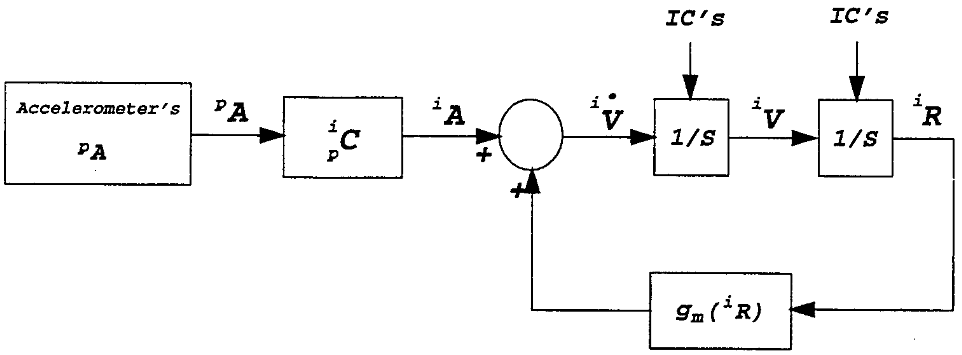


Figure 6.2: Calculation of Velocity and Position

Therefore:

$$^{i}\dot{V} = {}^{i}_{p}C\,{}^{p}A + {}^{i}g_m(R)$$

Notation

$$\begin{aligned} \frac{d}{dt}(\;)_i &= wrt\ \{i\} \\ \frac{d}{dt}(\;)_e &= wrt\ \{e\} \end{aligned}$$

Let R be position vector of the vehicle and Ω_{ei} be the earth's turning rate. Then by the coriolis theorem, (Equation 6.2):

$$\left(\frac{dR}{dt}\right)_i = \left(\frac{dR}{dt}\right)_e + \Omega_{ei} \times R = V + \Omega_{ei} \times R \tag{6.8}$$

Where

$$V = \left(\frac{dR}{dt}\right)_e$$

Differentiating Equation 6.8 *wrt* $\{i\}$ and remembering that Ω_{ei} is a constant, we get

$$\left(\frac{d^2R}{dt^2}\right)_i = \left(\frac{dV}{dt}\right)_i + \Omega_{ei} \times \left(\frac{dR}{dt}\right)_i \tag{6.9}$$

Now substitute Equation 6.8 into Equation 6.9 we get

$$\left(\frac{d^2R}{dt^2}\right)_i = \left(\frac{dV}{dt}\right)_i + \Omega_{ei} \times V + \Omega_{ei} \times (\Omega_{ei} \times R)$$

Now, using the Coriolis theorem for $\left(\frac{dV}{dt}\right)_i$:

$$\left(\frac{dV}{dt}\right)_i = \left(\frac{dV}{dt}\right)_p + \Omega_{pi} \times V$$

Where

$$\Omega_{pi} = \text{ angular velocity of } \{p\} \; wrt \; \{i\}$$

Equation 6.9 becomes

$$\begin{aligned}\left(\frac{d^2R}{dt^2}\right)_i &= \left(\frac{dV}{dt}\right)_p + \Omega_{pi} \times V + \Omega_{ei} \times V + \Omega_{ei} \times (\Omega_{ei} \times R) \\ &= \left(\frac{dV}{dt}\right)_p + (\Omega_{pi} + \Omega_{ei}) \times V + \Omega_{ei} \times (\Omega_{ei} \times R) \qquad (6.10)\end{aligned}$$

Since

$$\left(\frac{d^2R}{dt^2}\right)_i = A + g_m(R),$$

we re–arrange Equation 6.10 to obtain

$$A = \left(\frac{dV}{dt}\right)_p + (\Omega_{pi} + \Omega_{ei}) \times V + \Omega_{ei} \times (\Omega_{ei} \times R) - g_m(R) \qquad (6.11)$$

Now, let

$$\begin{aligned} g(R) &= g_m(R) - \Omega_{ei} \times (\Omega_{ei} \times R) \\ \text{which is } &\approx \sqrt{\frac{\|g(R)\|}{\|R\|}} R \Leftarrow \text{ Schuler frequency}\end{aligned}$$

Substituting into Equation 6.11, we get

$$\Rightarrow A = \left(\frac{dV}{dt}\right)_p + (\Omega_{pi} + \Omega_{ei}) \times V - g(R) \qquad (6.12)$$

Where

$$\Omega_{pi} = \Omega_{pe} + \Omega_{ei}$$

Then substituting into Equation 6.12

$$A = \left(\frac{dV}{dt}\right)_p + (\Omega_{pe} + 2\Omega_{ei}) \times V - g(R)$$

Rearranging terms

$$\left(\frac{dV}{dt}\right)_p = A - (\Omega_{pe} + 2\Omega_{ei}) \times V + g(R) \tag{6.13}$$

Since V is a vector, or

$$\Rightarrow V = \begin{bmatrix} V_x & V_y & V_z \end{bmatrix}^T$$

We can re–write Equation 6.13 [Ref. 1, p.246] as

$$\begin{cases} \dot{V}_x &= A_x - (\Omega_{pe_y} + 2\Omega_{eiy})V_z + (\Omega_{pe_z} + 2\Omega_{eiz})V_y + g_x \\ \dot{V}_y &= A_y - (\Omega_{pe_z} + 2\Omega_{eiz})V_x + (\Omega_{pe_x} + 2\Omega_{eix})V_z + g_y \\ \dot{V}_z &= A_z - (\Omega_{pe_x} + 2\Omega_{eix})V_y + (\Omega_{pe_y} + 2\Omega_{eiy})V_x + g_z \end{cases}$$

B. GRAVITATIONAL MODEL

- Assumes Earth's mass distribution is symmetric around Earth's z_e axis

- Gravitational potential in $\{e\}$

$$U(R,\phi) = -\frac{\mu}{R}(1 - \sum_{n=2}^{\infty} J_n(\frac{a}{R})^n P_n(\sin\phi))$$

based on reference ellipsoid

Where

μ = earth's gravitational constant

a = mean equatorial radius of earth

R = $\sqrt{x_e^2 + y_e^2 + z_e^2}$

ϕ = latitude ($\sin\phi = \frac{z_e}{R}$)

J_n = coefficients of zonal harmonics (constants determined from observations of orbit perturbations of artificial satellites)

$P_n(\sin\phi)$ = associated Legendre Polynomial of the 1st kind

$\frac{\mu}{R}$ = potential of spherical mass

The rest account for:

- * Earth is bulged at the equator

- * Flattened at the poles

- * Generally asymmetric

<u>Note</u>, if symmetry *wrt* equator is assumed ⇒ odd harmonics vanish.

$$\text{gravity vector} \Rightarrow g(R) = \begin{bmatrix} g_x & g_y & g_z \end{bmatrix}^T$$

$$g_x = \frac{\partial U}{\partial x},\ g_y = \frac{\partial U}{\partial y},\ g_z = \frac{\partial U}{\partial z}$$

$g(R)$ in ECI is given in [Ref. 1, p. 147].

<u>Formula for Computing Vehicle's Position & Velocity in $\{n\}$</u>

$$\left[\frac{dV}{dt}\right]_p = A - (\Omega_{ne} + 2\Omega_{ei}) \times V + g(R)$$

Compute in $\{n\} = ENU$. Then

$$^n\Omega_{ei} = \begin{bmatrix} \Omega_E \\ \Omega_N \\ \Omega_U \end{bmatrix} = \begin{bmatrix} 0 \\ \Omega \cos\phi \\ \Omega \sin\phi \end{bmatrix}$$

$$^n\Omega_{ne} = \begin{bmatrix} \omega_E \\ \omega_N \\ \omega_U \end{bmatrix} = \begin{bmatrix} -\frac{V_N}{R_\phi + h} \\ \frac{V_E}{R_\lambda + h} \\ \frac{V_E}{R_\lambda + h}\tan\phi \end{bmatrix},\ \text{where } {}^nV = \begin{bmatrix} V_E \\ V_N \\ V_U \end{bmatrix} = \begin{bmatrix} V_x \\ V_y \\ V_z \end{bmatrix}$$

Also

$$^n\Omega_{ne} = \begin{bmatrix} -\dot\phi \\ \dot\lambda \cos\phi \\ \dot\lambda \sin\phi \end{bmatrix}$$

Therefore:

$$\left[\frac{d\,{}^{n}V}{dt}\right]_p = {}^{n}A - ({}^{n}\Omega_{ne} + 2\,{}^{n}\Omega_{ei}) \times {}^{n}V + {}^{n}g(R) \qquad (6.14)$$

from pg.152

$$\frac{d}{dt}\begin{bmatrix} \dot{V}_E \\ \dot{V}_N \\ \dot{V}_U \end{bmatrix} = \begin{bmatrix} A_E \\ A_N \\ A_U \end{bmatrix} - \begin{bmatrix} 0 & -\omega_U - 2\Omega_E & \omega_N + 2\Omega_N \\ \omega_U + 2\Omega_U & 0 & -\omega_E - 2\Omega_E \\ -\omega_N - 2\Omega_N & \omega_E + 2\Omega_E & 0 \end{bmatrix}\begin{bmatrix} V_E \\ V_N \\ V_U \end{bmatrix} + \begin{bmatrix} g_x \\ g_y \\ g_z \end{bmatrix}_n$$

$$\begin{aligned} V_E &= \int_0^t \dot{V}_E\,dt \\ \Rightarrow\ V_N &= \int_0^t \dot{V}_N\,dt \\ V_U &= \int_0^t \dot{V}_U\,dt \end{aligned}$$

In $\{n\}$ for spherical earth:

$${}^{n}g_m(R) \simeq {}^{n}g_1(R) \simeq \begin{bmatrix} 0 \\ 0 \\ g_{mo}(\frac{R_o}{R})^2 \end{bmatrix} \simeq \begin{bmatrix} 0 \\ 0 \\ g_{mo}(1 - \frac{2h}{R_o}) \end{bmatrix}$$

Finally, note:

$$\begin{aligned} \dot{\phi} &= \frac{V_N}{R_\phi + h} \\ \dot{\lambda} &= \frac{V_E}{(R_\lambda + h)\cos\phi} \end{aligned}$$

Figure 6.3 shows the standard mechanization for Equation 6.14.

Wander Azimuth Mechanization

Let

$$V = \left(\frac{dR}{dt}\right)_e$$

$$\left[\frac{d\,{}^{p}V}{dt}\right]_p = {}^{p}A - ({}^{p}\Omega_{pe} + 2\,{}^{p}\Omega_{ei}) \times {}^{p}V + {}^{p}g(R)$$

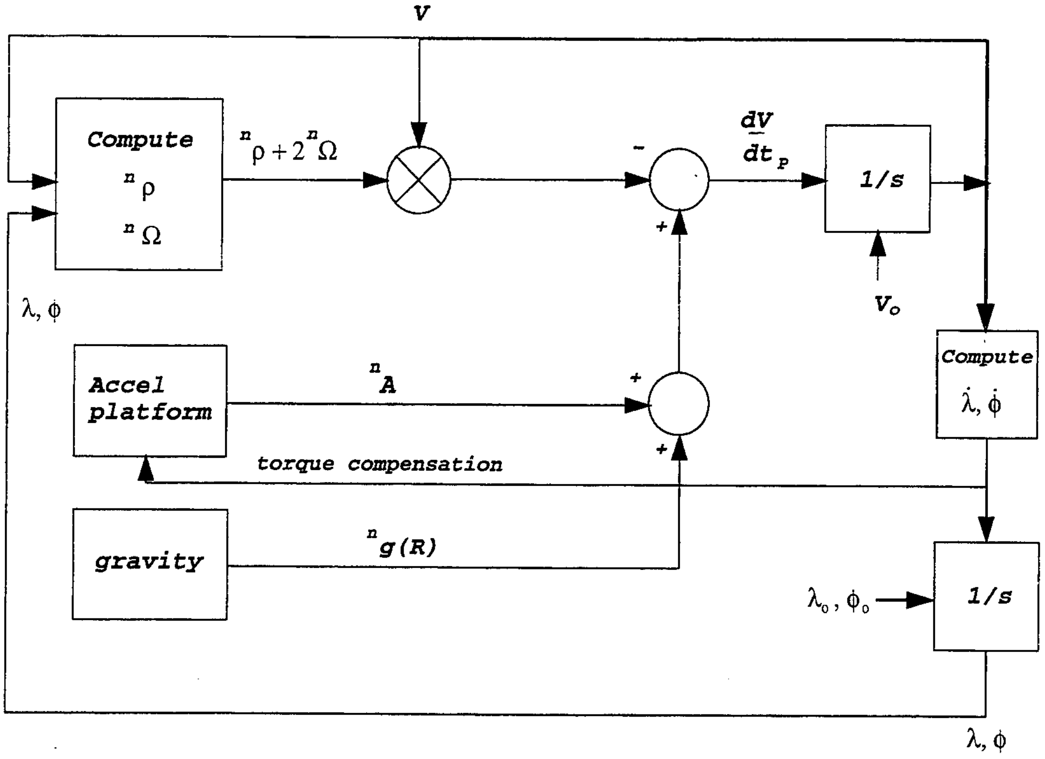


Figure 6.3: Standard Mechanization

Recall for $\{n\} = ENU$

$$^n\Omega_{ne} = \begin{bmatrix} -\frac{V_N}{R_\phi+h} \\ \frac{V_E}{R_\lambda+h} \\ \frac{V_E}{R_\lambda+h}\tan\phi \end{bmatrix} = \begin{bmatrix} -\dot\phi \\ \dot\lambda\cos\phi \\ \dot\lambda\sin\phi \end{bmatrix}$$

$$^n\Omega_{ei} = \begin{bmatrix} 0 \\ \Omega\cos\phi \\ \Omega\sin\phi \end{bmatrix}$$

Note

$$^p_nC = \begin{bmatrix} c\alpha & s\alpha & 0 \\ -s\alpha & c\alpha & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

$$\begin{aligned} \Rightarrow\ ^p\Omega_{pe} &= \ ^p\Omega_{pn} + ^p_nC\ ^n\Omega_{ne} \\ &= \begin{bmatrix} 0 \\ 0 \\ \dot\alpha \end{bmatrix} + \begin{bmatrix} c\alpha & s\alpha & 0 \\ -s\alpha & c\alpha & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} -\dot\phi \\ \dot\lambda c\phi \\ \dot\lambda s\phi \end{bmatrix} \end{aligned}$$

$$\begin{aligned}
&= \begin{bmatrix} -\dot{\phi}c\alpha + \dot{\lambda}c\phi s\alpha \\ \dot{\phi}s\alpha + \dot{\lambda}c\phi c\alpha \\ \dot{\alpha} + \dot{\lambda}s\phi \end{bmatrix} \\
{}^p\Omega_{ei} &= \begin{bmatrix} c\alpha & s\alpha & 0 \\ -s\alpha & c\alpha & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 0 \\ \Omega\cos\phi \\ \Omega\sin\phi \end{bmatrix} \\
&= \begin{bmatrix} \Omega c\phi s\alpha \\ \Omega c\phi c\alpha \\ \Omega s\phi \end{bmatrix} \\
{}^pV &= \begin{bmatrix} c\alpha & s\alpha & 0 \\ -s\alpha & c\alpha & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} V_E \\ V_N \\ V_U \end{bmatrix} \\
&= \begin{bmatrix} {}^pV_x \\ {}^pV_y \\ {}^pV_z \end{bmatrix} \\
\Rightarrow \begin{bmatrix} V_E \\ V_N \\ V_U \end{bmatrix} &= \begin{bmatrix} c\alpha & s\alpha & 0 \\ -s\alpha & c\alpha & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} {}^pV_x \\ {}^pV_y \\ {}^pV_z \end{bmatrix}
\end{aligned}$$

And

$$\begin{aligned}
\dot{\lambda} &= \frac{V_N}{R_\phi + h} \\
\dot{\phi} &= \frac{V_E}{(R_\lambda + h)c\phi} \\
\dot{\alpha} &= -\dot{\lambda}\sin\phi \text{ (wander azimuth)}
\end{aligned}$$

Alternatively, compute λ, ϕ, and α from direction cosines:

$${}^p_e\dot{C} = {}^p_eC\,S\,({}^p\Omega_{ne}) \tag{6.15}$$

Integrate Equation 6.15 to get ${}^p_cC$. Then, for $\{n\} = ENU$:

$${}^p_eC = \begin{bmatrix} c\alpha c\lambda - s\alpha s\phi s\lambda & s\alpha c\phi & -c\alpha s\lambda - s\alpha s\phi c\lambda \\ -s\alpha c\lambda - c\alpha s\phi s\lambda & c\alpha c\phi & s\alpha s\lambda - c\alpha s\phi c\lambda \\ c\phi s\lambda & s\phi & c\phi c\lambda \end{bmatrix}$$

Where

$${}^p_eC = \begin{bmatrix} C_{11} & C_{12} & C_{13} \\ C_{21} & C_{22} & C_{23} \\ C_{31} & C_{32} & C_{33} \end{bmatrix}$$

And

$$\begin{aligned}\phi &= \sin^{-1} C_{32} \\ \lambda &= \tan^{-1} \frac{C_{31}}{C_{33}} \\ \alpha &= \tan^{-1} \frac{C_{12}}{C_{22}}\end{aligned}$$

Wander Azimuth; North Slaved

Assume elliptical earth (geodetic)

$$R_\phi = R_o(1 - 2f + 2f \sin^2 \phi)$$

$$R_\lambda = R_o(1 + f \sin^2 \phi), \text{ where } f = \frac{1}{2}(1 - \frac{b^2}{a^2})$$

Note, the spherical earth ($R_\phi = R_\lambda = R_o$) and $f = 0$.

Recall

$$^n\Omega_{ne} = \begin{bmatrix} -\frac{V_N}{R_\phi+h} \\ \frac{V_E}{R_\lambda+h} \\ \frac{V_E}{R_\lambda+h} \tan \phi \end{bmatrix} = \begin{bmatrix} -\dot{\phi} \\ \dot{\lambda} c\phi \\ \dot{\lambda} s\phi \end{bmatrix}$$

1. Compute torquing rates

$$^p\Omega_{pi} = {}^p\Omega_{pe} + {}^p\Omega_{ei}$$

2. Compute $^p\Omega_{pe}$

$$\begin{aligned} ^p\Omega_{pe} &= {}^p\Omega_{ne} + {}^p\Omega_{pn} \\ &= {}^p_nC\, {}^n\Omega_{ne} + {}^p_nC \begin{bmatrix} 0 \\ 0 \\ \dot{\alpha} \end{bmatrix} \\ &= \begin{bmatrix} -\dot{\phi} c\alpha + \dot{\lambda} c\phi s\alpha \\ \dot{\phi} s\alpha + \dot{\lambda} c\phi c\alpha \\ \dot{\lambda} s\phi + \dot{\alpha} \end{bmatrix} \end{aligned}$$

Also,

$$^n\Omega_{ne} = \begin{bmatrix} -\frac{V_N}{R_\phi+h} \\ \frac{V_E}{R_\lambda+h} \\ \frac{V_E}{R_\lambda+h} \tan \phi \end{bmatrix} \leftarrow \text{ torquing rate blows up near poles (for North Slaved)}$$

Now wander–azimuth

$$
{}^p\Omega_{pe_z} = 0 \Rightarrow \dot{\alpha} = -\dot{\lambda} s\phi
$$

$$
\Rightarrow {}^p\Omega_{pe} = \begin{bmatrix} -\dot{\phi} c\alpha + \dot{\lambda} c\phi s\alpha \\ \dot{\phi} s\alpha + \dot{\lambda} c\phi c\alpha \\ \dot{\lambda} s\phi + \dot{\alpha} = 0 \end{bmatrix}
$$

3.

$$
\left[\frac{d\,{}^pV}{dt}\right]_p = {}^pA - ({}^p\Omega_{pe} + 2\,{}^p\Omega_{ei}) \times {}^pV + g(R)
$$

4.

$$
\begin{aligned}
\dot{\phi} &= \frac{V_N}{R_\phi + h} \\
\dot{\lambda} &= \frac{V_E}{(R_\lambda + h)c\phi} \\
\dot{\alpha} &= -\dot{\lambda} s\phi \text{ (for wander–azimuth).}
\end{aligned}
$$

5. Compute h

$$
{}^eR = \begin{bmatrix} x_e \\ y_e \\ z_e \end{bmatrix} \begin{bmatrix} (R_\lambda + h)c\phi c\lambda \\ (R_\lambda + h)c\phi s\lambda \\ (R_\lambda(1-\varepsilon^2) + h)s\phi \end{bmatrix}
$$

$$
\Rightarrow h = \frac{z_e}{s\phi} - (1-\varepsilon^2)R_\lambda,\ 1-\varepsilon^2 = \frac{b^2}{a^2},\ \varepsilon^2 = 2f
$$

Where

$$
\left[\frac{d}{dt}\,{}^eR\right]_e = {}^e_nC\,{}^nV = {}^e_pC\,{}^pV
$$

C. SPACE STABILIZED MECHANIZATION

* The accelerometer platform is always aligned with $\{i\}$ (maintain constant orientation)

* Assume spherical earth

* Torquing: $\omega_x = \omega_y = \omega_z = 0$

* Governing equation:

$${}^{i}R = {}^{i}_{p}C\,{}^{p}A + {}^{i}g = {}^{i}_{p}C({}^{p}A + {}^{p}g)$$

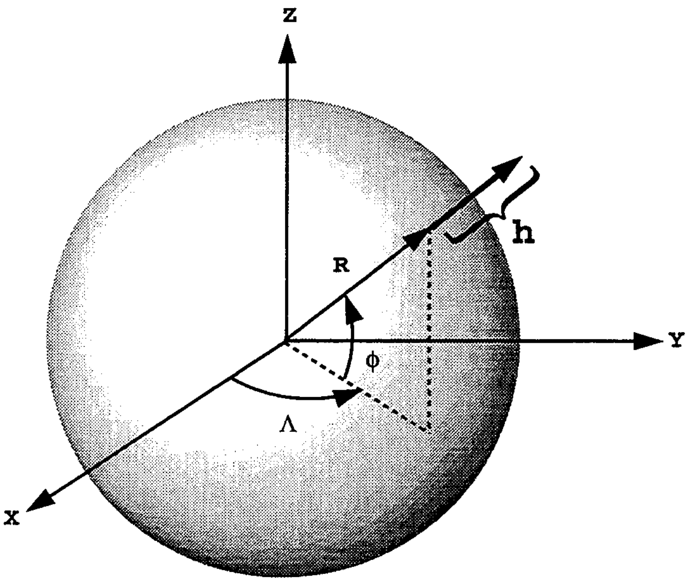


Figure 6.4: Latitude and Longitude in $\{i\}$ Frame

Therefore

$${}^{i}R = \begin{bmatrix} (R+h)c\phi c\Lambda \\ (R+h)c\phi s\Lambda \\ (R+h)s\phi \end{bmatrix}$$

$$\Rightarrow \Lambda = \tan^{-1}\left(\frac{{}^{i}R_y}{{}^{i}R_x}\right) = \lambda - \lambda_o + \Omega t$$

$$\Rightarrow \lambda = \lambda_o + \tan^{-1}\left(\frac{{}^{i}R_y}{{}^{i}R_x}\right) + \Omega t$$

$$\phi = \sin^{-1}\left(\frac{{}^{i}R_z}{{}^{i}R + h}\right)$$

$${}^{n}V. = {}^{n}\left(\frac{dR}{dt}\right)_e$$

Recall

$${}^{i}\dot{R} = \left(\frac{d^{i}R}{dt}\right)_i = \left(\frac{d^{i}R}{dt}\right)_e + {}^{i}\Omega_{ei} \times {}^{i}R$$

Then

$${}^nV = {}^n_iC\left(\frac{dR}{dt}\right)_e = {}^n_iC({}^i\dot{R} - {}^i\Omega \times {}^iR)$$

* Compute ${}^nV$:

$$\begin{aligned}
{}^n_iC &= \begin{bmatrix} c\phi & 0 & s\phi \\ 0 & 1 & 0 \\ -s\phi & 0 & c\phi \end{bmatrix}\begin{bmatrix} c\Lambda & s\Lambda & 0 \\ -s\Lambda & c\Lambda & 0 \\ 0 & 0 & 1 \end{bmatrix} \\
&= \begin{bmatrix} c\phi c\Lambda & c\phi s\Lambda & s\phi \\ -s\Lambda & c\Lambda & 0 \\ -s\phi c\Lambda & -s\phi s\Lambda & c\phi \end{bmatrix}
\end{aligned}$$

* Finally, ${}^i_pC$ = const = initial values

* ${}^ig = {}^i_nC\,{}^ng$; ${}^i_nC = {}^n_iC^T$

$${}^ng = \begin{bmatrix} 0 \\ 0 \\ g_z \end{bmatrix}$$

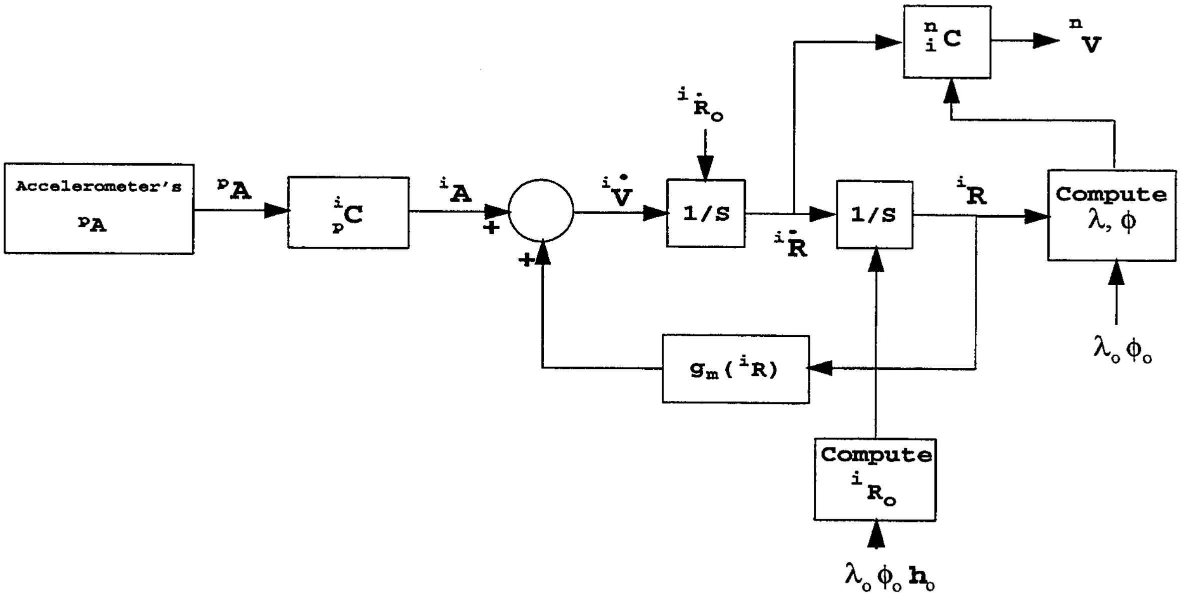


Figure 6.5: Navigation Mechanization ${}^pA \rightarrow {}^nV$

* Assume ellipsoidal earth

- * Torquing ⇒ the same

- * Governing equation:

$$\begin{aligned}
{}^{i}\ddot{R} &= {}^{i}_{p}C\,{}^{p}A + {}^{i}g \\
{}^{i}R &= \begin{bmatrix} (R_\lambda + h)c\phi c\lambda \\ (R_\lambda + h)c\phi s\lambda \\ \left[(1-\varepsilon^2)R_\lambda + h\right] s\phi \end{bmatrix} \\
\Lambda &= \tan^{-1}\left(\frac{{}^{i}R_y}{{}^{i}R_x}\right) \\
\lambda &= \lambda_o + \Lambda - \Omega t \\
\phi &= \sin^{-1}\left(\frac{{}^{i}R_z}{(1-\varepsilon^2)R_\lambda + h}\right)
\end{aligned}$$

or consider

$$({}^{i}R_x^2 + {}^{i}R_y^2)^{1/2} = \left((R_\lambda + h)^2\cos^2\phi\right)^{1/2} = (R_\lambda + h)\cos\phi$$

$$\Rightarrow \tan\phi = \frac{{}^{i}R_z}{\sqrt{{}^{i}R_x^2 + {}^{i}R_y^2}}$$

Example Wander Azimuth Navigation

$\{n\} = UEN$

Vehicle is traveling in a plane defined by $\phi_o = 45^\cdot$ with the equitorial plane. Vehicle's velocity V along it's path is $400\frac{nmi}{hr}$. Vehicle has wander azimuth system aboard.

1. What is ${}^{n}V$?

$${}^{n}V = \begin{bmatrix} 0 \\ 400 \\ 0 \end{bmatrix} = \begin{bmatrix} 0 \\ 178\frac{m}{sec} \\ 0 \end{bmatrix}$$

2. What is ${}^{n}\Omega_{ne}$, ${}^{n}\Omega_{ei}$, ${}^{n}\Omega_{ni}$

$${}^{e}\Omega_{ne} = \begin{bmatrix} 0 \\ 0 \\ \frac{V}{(R\cos\phi_o)} \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ \dot{\lambda} \end{bmatrix}$$

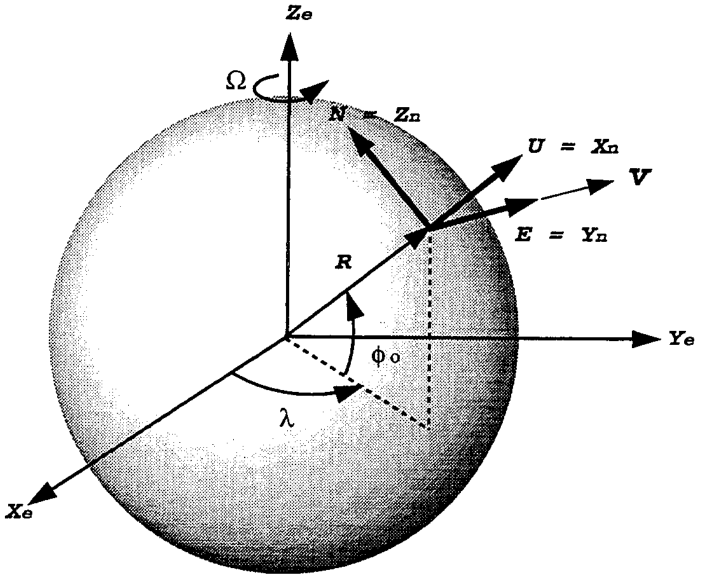


Figure 6.6: Wander Azimuth Navigation

Now

$$
\begin{aligned}
{}^{n}\Omega_{ne} &= \begin{bmatrix} c\phi_o & 0 & s\phi_o \\ 0 & 1 & 0 \\ -s\phi_o & 0 & c\phi_o \end{bmatrix} \begin{bmatrix} c\lambda & s\lambda & 0 \\ -s\lambda & c\lambda & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 0 \\ 0 \\ \dot{\lambda} \end{bmatrix} \\
&= \begin{bmatrix} \dot{\lambda} s\phi_o \\ 0 \\ \dot{\lambda} c\phi_o \end{bmatrix} \\
&= \begin{bmatrix} \frac{V}{R}\tan\phi_o \\ 0 \\ \frac{V}{R} \end{bmatrix}
\end{aligned}
$$

Check Figure 6.7

and solve for

$$
{}^{n}\Omega_{ei} = \begin{bmatrix} \Omega s\phi_o \\ 0 \\ \Omega c\phi_o \end{bmatrix}
$$

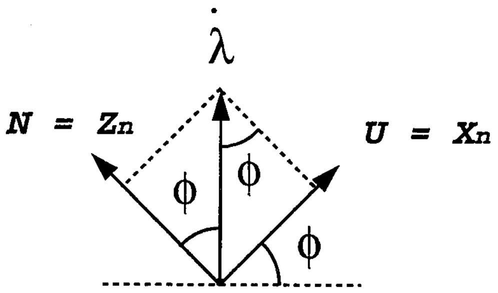


Figure 6.7: U and N Transformation

$$
{}^n\Omega_{ni} = \begin{bmatrix} \lambda s\phi_o + \Omega s\phi_o \\ 0 \\ \dot{\lambda} c\phi_o + \Omega c\phi_o \end{bmatrix} = \begin{bmatrix} \frac{V}{R}\tan\phi_o + \Omega s\phi_o \\ 0 \\ \frac{V}{R} + \Omega c\phi_o \end{bmatrix}
$$

3. What is ${}^p\Omega_{pi}$?

$$
\begin{aligned}
{}^p\Omega_{pi} &= {}^p_nC(\,{}^n\Omega_{ni} + {}^n\Omega_{pn}) \\
&= \begin{bmatrix} 1 & 0 & 0 \\ 0 & c\alpha & s\alpha \\ 0 & -s\alpha & c\alpha \end{bmatrix} \begin{bmatrix} \frac{V}{R}\tan\phi_o + \Omega s\phi_o + \dot{\alpha} \\ 0 \\ \frac{V}{R} + \Omega c\phi_o \end{bmatrix} \\
&= \begin{bmatrix} \frac{V}{R}\tan\phi_o + \Omega s\phi_o + \dot{\alpha} \\ (\frac{V}{R} + \Omega c\phi_o) s\alpha \\ (\frac{V}{R} + \Omega c\phi_o) c\alpha \end{bmatrix}
\end{aligned}
$$

For wander azimuth:

$$
\begin{aligned}
\dot{\alpha} &= -\dot{\lambda} s\phi_o \\
&= -\frac{V}{R}\tan\phi_o
\end{aligned}
$$

Therefore:

$$
\begin{aligned}
\phi &= \text{const} = \phi_o \\
\lambda(t) &= \int_{t_o}^{t} \frac{V}{Rc\phi_o}\,dt = \frac{V}{Rc\phi_o}(t - t_o) + \lambda(t_o) \\
\alpha(t) &= \int_{t_o}^{t} \frac{V}{R}\tan\phi_o\,dt = -\frac{V}{R}\tan\phi_o(t - t_o) + \alpha_o(t_o)
\end{aligned}
$$

Platform torquing:

$$\begin{cases} \omega_{xc} &= \Omega s\phi_o \leftarrow \text{ wander azimuth} \\ \omega_{yc} &= (\frac{V}{R} + \Omega c\phi_o)s\alpha \\ \omega_{zc} &= (\frac{V}{R} + \Omega s\phi_o)s\alpha \end{cases}$$

4. What do accelerometers measure?

$$\begin{aligned} {}^p\left(\frac{d^2R}{dt^2}\right)_i &= {}^pA + {}^pg_m(R), \\ {}^pg_m(R) &= \begin{bmatrix} -\frac{\mu}{R^2} \\ 0 \\ 0 \end{bmatrix} = \begin{bmatrix} -\frac{\mu}{(R_o+h)^2} \\ 0 \\ 0 \end{bmatrix} \end{aligned}$$

5. What is $\left(\frac{d^2R}{dt^2}\right)_i$ in terms of ${}^nV$, ${}^p\Omega_{ne}$, ${}^p\Omega_{ei}$?

Recall

$$\begin{aligned} \left(\frac{dR}{dt}\right)_i &= \left(\frac{dR}{dt}\right)_e + \Omega_{ei} \times R \\ &= V + \Omega_{ei} \times R \end{aligned}$$

Differentiating:

$$\left(\frac{d^2R}{dt^2}\right)_i = \left(\frac{dV}{dt}\right)_i + \Omega_{ei} \times (V + \Omega_{ei} \times R)$$

Applying the Coriolis theorem to $\left(\frac{dV}{dt}\right)_i$ and expanding the cross product terms, we get:

$$\begin{aligned} \left(\frac{d^2R}{dt^2}\right)_i &= \left(\frac{dV}{dt}\right)_p + \Omega_{pi} \times V + \Omega_{ei} \times V + \Omega_{ei} \times (\Omega_{ei} \times R) \\ &= \left(\frac{dV}{dt}\right)_p + (\Omega_{pe} + \Omega_{ei}) \times V + \Omega_{ei} \times V + \Omega_{ei} \times (\Omega_{ei} \times R) \\ &= \left(\frac{dV}{dt}\right)_p + (\Omega_{pe} + 2\Omega_{ei}) \times V + \Omega_{ei} \times (\Omega_{ei} \times V) \\ &= \left(\frac{dV}{dt}\right)_p + (\Omega_{pe} + 2\Omega_{ei}) \times V + \Omega_{ei} \times (\Omega_{ei} \times V), \text{ where } \Omega_{ei} \approx 0 \\ &\simeq (\Omega_{pe} + 2\Omega_{ei}) \times V + \left(\frac{dV}{dt}\right)_p \qquad (6.16) \end{aligned}$$

Apply the Coriolis theorem to $\left(\frac{dV}{dt}\right)_e$:

$$\left(\frac{dV}{dt}\right)_e = \left(\frac{dV}{dt}\right)_p + \Omega_{pe} \times V$$

Solving for $\left(\frac{dV}{dt}\right)_p$ and remembering that $\left(\frac{dV}{dt}\right)_e = 0$:

$$\left(\frac{dV}{dt}\right)_p = -\Omega_{pe} \times V \tag{6.17}$$

Substituting 6.17 into 6.16 we get:

$$\left(\frac{d^2R}{dt^2}\right)_i = -\Omega_{pe} \times V + (\Omega_{pe} + 2\Omega_{ei}) \times V \tag{6.18}$$

Therefore,

$$\begin{aligned}
{}^p\!\left(\frac{d^2R}{dt^2}\right)_i &= ({}^p\Omega_{pe} + 2\,{}^p\Omega_{ei}) \times {}^pV \\
&\simeq \left(\begin{bmatrix} \Omega s\phi_o \\ (\frac{V}{R} + \Omega c\phi_o)s\alpha \\ (\frac{V}{R} + \Omega c\phi_o)c\alpha \end{bmatrix} + 2 \begin{bmatrix} \Omega s\phi_o \\ \Omega c\phi_o s\alpha \\ \Omega c\phi_o c\alpha \end{bmatrix} \right) \times \left(\begin{bmatrix} 1 & 0 & 0 \\ 0 & c\alpha & s\alpha \\ 0 & -s\alpha & c\alpha \end{bmatrix} \begin{bmatrix} 0 \\ V \\ 0 \end{bmatrix} \right) \\
&= \begin{bmatrix} 3\Omega s\phi_o \\ \frac{V}{R}s\alpha + 3\Omega c\phi_o s\alpha \\ \frac{V}{R}c\alpha + 3\Omega c\phi_o c\alpha \end{bmatrix} \times \begin{bmatrix} 0 \\ Vc\alpha \\ -Vs\alpha \end{bmatrix}
\end{aligned}$$

Note, $\Omega = 7.5 \times 10^{-7}\ \frac{rad}{sec}$

$$\Rightarrow {}^p\!\left(\frac{d^2R}{dt^2}\right)_i \approx \underbrace{\begin{bmatrix} 0 \\ \frac{V}{R}s\alpha \\ \frac{V}{R}c\alpha \end{bmatrix}}_{A} \times \underbrace{\begin{bmatrix} 0 \\ Vc\alpha \\ -Vs\alpha \end{bmatrix}}_{B}$$

Making "A" a skew symmetric matrix and applying $A \times B = S(A)B$, we get:

$$\begin{aligned}
{}^p\!\left(\frac{d^2R}{dt^2}\right)_i &= \begin{bmatrix} 0 & -\frac{V}{R}c\alpha & \frac{V}{R}s\alpha \\ \frac{V}{R}c\alpha & 0 & 0 \\ -\frac{V}{R}s\alpha & 0 & 0 \end{bmatrix} \begin{bmatrix} 0 \\ Vc\alpha \\ -Vs\alpha \end{bmatrix} \\
&= \begin{bmatrix} -\frac{V^2}{R}c^2\alpha - \frac{V^2}{R}s^2\alpha \\ 0 \\ 0 \end{bmatrix} = \begin{bmatrix} -\frac{V^2}{R} \\ 0 \\ 0 \end{bmatrix} = {}^pA + {}^pg(R) \\
&= \begin{bmatrix} a_x \\ a_y \\ a_z \end{bmatrix} + \begin{bmatrix} g_x \\ 0 \\ 0 \end{bmatrix} \Rightarrow \left\{ \begin{array}{lcl} a_x &=& -\frac{V^2}{R} - g_x \\ a_y &=& 0 \\ a_z &=& 0 \end{array} \right.
\end{aligned}$$

Recall from classical physics ⇒ centrifugal acceleration = $\frac{V^2}{R}$

Therefore,

$$a_x = -\frac{178^2}{6.4 \times 10^6} - 9.8 \approx -9.8 \frac{m}{sec^2}$$

Where I assumed $R = R_o = 6400\ km$

D. ERROR ANALYSIS

* Idea: Perturb the navigation equations around the true values of the position and velocity vectors. Consider tangent plane mechanization, Figure 6.8.

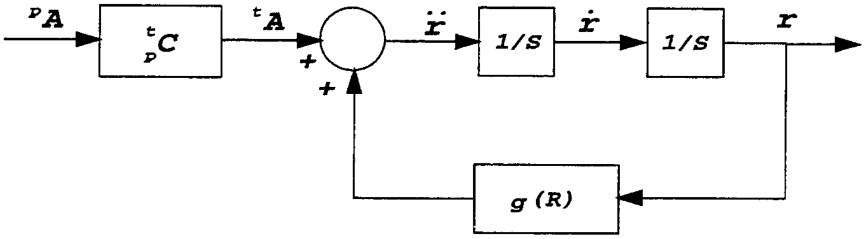


Figure 6.8: Tangent Plane Mechanization

Let $V = \dot{r}$

Then $\dot{V} = A + g(R)$

$$\Rightarrow \begin{cases} \dot{r} &= V \\ \dot{V} &= g(R) + A \end{cases}$$

From Figure 6.9 $R = {}^tR = {}^tR_{EC} + {}^tr$

$$\|R\| = \sqrt{x^2 + y^2 + (z + R_o)^2}$$

$$g = \frac{\mu}{\|R\|^2} i_p = \frac{\mu}{\|R\|^2} \begin{bmatrix} -\frac{x}{\|R\|} \\ -\frac{y}{\|R\|} \\ -\frac{(z+R_o)}{\|R\|} \end{bmatrix}$$

Therefore we obtain

$$\begin{cases} \dot{r} &= V \\ \dot{V} &= A + \begin{bmatrix} -\frac{\mu x}{\|R\|^3} \\ -\frac{\mu y}{\|R\|^3} \\ -\frac{\mu(z+R_o)}{\|R\|^3} \end{bmatrix} \end{cases} \quad (*)$$

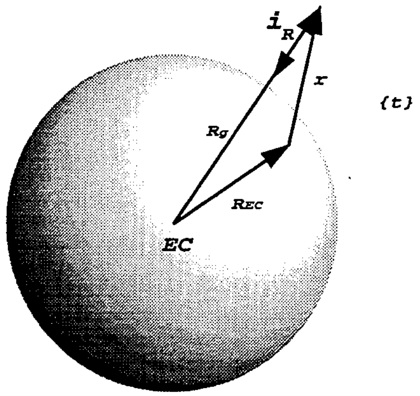


Figure 6.9: R_{EC}

Now, suppose ${}^t_pC$ is known exactly but,

$$\begin{array}{rcl} r & = & r_o + \delta r \\ V & = & V_o + \delta V \\ {}^tA & = & A_o + \delta A \end{array} \qquad (6.19)$$

Where r_o, V_o, A_o are true values and δ's are small errors. Now let's linearize 6.19 around r_o, V_o, A_o to obtain a linear model of the error dynamics.

$$\Rightarrow \left\{ \begin{array}{rcl} \dot{\delta r} & = & \delta V \\ \delta \dot{V} & = & \delta A + \left[\begin{array}{ccc} \frac{\partial g(R)}{\partial x} & \frac{\partial g(R)}{\partial y} & \frac{\partial g(R)}{\partial z} \end{array} \right] \begin{bmatrix} \delta x \\ \delta y \\ \delta z \end{bmatrix} \end{array} \right.$$

Where

$$\delta r = \begin{bmatrix} \delta x \\ \delta y \\ \delta z \end{bmatrix}, \ \delta V = \begin{bmatrix} \delta V_x \\ \delta V_y \\ \delta V_z \end{bmatrix}, \ \delta A = \begin{bmatrix} \delta A_x \\ \delta A_y \\ \delta A_z \end{bmatrix}$$

Now

$$\frac{\partial}{\partial x} \begin{bmatrix} -\frac{\mu x}{||R||^3} \\ -\frac{\mu y}{||R||^3} \\ -\frac{\mu(z+R_o)}{||R||^3} \end{bmatrix} \simeq \begin{bmatrix} -\frac{g_o}{R_o} \\ 0 \\ 0 \end{bmatrix}, \text{ same for} \frac{\partial}{\partial y} g(R)$$

Consider

$$\frac{\partial}{\partial z} g(R) \approx \begin{bmatrix} 0 \\ 0 \\ -\frac{\partial}{\partial z} \mu \frac{z+R_o}{||R||^3} \end{bmatrix} \simeq \begin{bmatrix} 0 \\ 0 \\ \frac{2g_o}{R_o} \end{bmatrix}$$

Where

$$\begin{aligned}
\mu\frac{\partial}{\partial z}\left(\frac{z+R_o}{\|R\|^3}\right) &= \mu\frac{\|R\|^3-(z+R_o)3\|R\|^2\frac{1}{2}2(z+R_o)/\|R\|}{\|R\|^6} \\
&= \mu\frac{\|R\|^3-3(z+R_o)^2\|R\|}{\|R\|^6} \\
&\approx -\frac{2\mu}{R_o^3} \simeq -2\frac{g_o}{R_o}
\end{aligned}$$

and

$$\begin{aligned}
\frac{\partial}{\partial z}\|R\| &= \frac{\partial}{\partial z}\sqrt{x^2+y^2+(z+R_o)^2} \\
&= \frac{1}{2}\frac{1}{\sqrt{x^2+y^2+(z+R_o)^2}}2(z+R_o) = \frac{z+R_o}{\|R\|}
\end{aligned}$$

Therefore:

$$\left\{\begin{array}{lcl}
\delta\dot{r}_x & = & \delta V_x \\
\delta\dot{r}_y & = & \delta V_y \\
\delta\dot{r}_z & = & \delta V_z \\
\delta\dot{V}_x & = & -\frac{g_o}{R_o}\delta r_x + \delta A_x = \ddot{r}_x \\
\delta\dot{V}_y & = & -\frac{g_o}{R_o}\delta r_y + \delta A_y = \ddot{r}_y \\
\delta\dot{V}_z & = & 2\frac{g_o}{R_o}\delta r_z + \delta A_z = \ddot{r}_z
\end{array}\right. \qquad (6.20)$$

$$\Rightarrow\left\{\begin{array}{lcl}
\ddot{r}_x & = & -\frac{g_o}{R_o}\delta r_x + \delta A_x \Rightarrow \text{ sinusoid} \\
\ddot{r}_y & = & -\frac{g_o}{R_o}\delta r_y + \delta A_y \Rightarrow \text{ sinusoid} \\
\ddot{r}_z & = & 2\frac{g_o}{R_o}\delta r_z + \delta A_x \Rightarrow \text{ hyperbola}
\end{array}\right.$$

We have shown INS equations have the form

$$\dot{x} = f(x, u) \qquad (6.21)$$

where

x = state vector consisting of vehicle position and velocity

u = accelerometer inputs

Now one can be used to obtain error equations for INS mechanization.

Let

$$\begin{aligned} x &= x_o + \delta x \\ u &= u_o + \delta u \end{aligned}$$

where x_o, u_o are true values and $\delta x, \delta u$ are errors.

Then

$$\begin{aligned} \dot{\delta x} &= \left.\frac{\partial f}{\partial x}\right|_0 \delta x + \left.\frac{\partial f}{\partial u}\right|_0 \delta u \\ &= A\delta x + B\delta u \end{aligned} \tag{6.22}$$

Equation 6.22 describes error dynamics of the navigation equation 6.21. Now, let's consider the vertical channel equation in the tangent plane. The mechanization is shown in Figure 6.10.

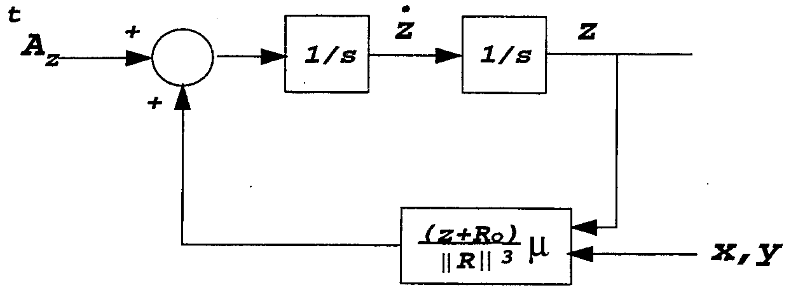


Figure 6.10: Tangent Mechanization: Vertical Channel

where

$$||R|| = \sqrt{x^2 + y^2 + (z + R_o)^2}$$

or

$$\begin{cases} \dot{z} = v_z \\ \dot{v}_z = \frac{(z+R_o)}{||R||^3} + A_z \end{cases} \tag{6.23}$$

Now let

$$z = z_o + \delta z,\ v_z = v_{zo} + \delta v_z,\ A_z = A_{zo} + \delta A_z$$

then from Equation 6.20 we obtain:

$$\begin{cases} \dot{\delta z} &= \delta v_z \\ \delta \dot{v}_z &= \frac{2g_o}{R_o}\delta z + \delta A_z \end{cases}$$

The mechanization for the vertical channel error equations is shown in Figure 6.11,

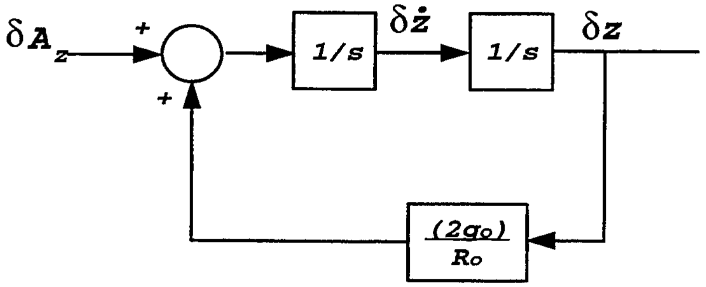


Figure 6.11: Vertical Channel with Error

$$\frac{\delta z(s)}{\delta A_z(s)} = \frac{1}{s^2 + \frac{2g_o}{R_o}}$$

where the poles are $\pm\sqrt{\frac{2g_o}{R_o}}$ which show that the errors in the vertical channel blow up. Therefore, let's use a complementary filter to stabilize the errors, Figure 6.12.

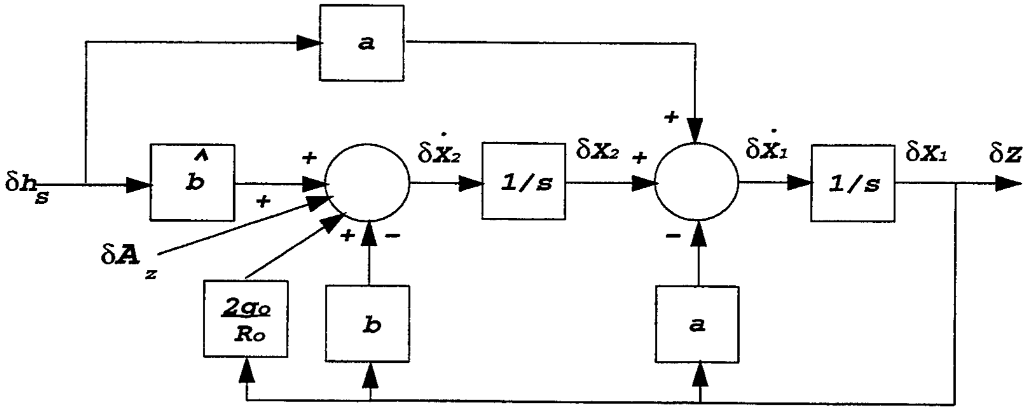


Figure 6.12: Vertical Channel Complementary Filter

Here $\delta\, h_s$ represents changes in h_s. Complementary filter equations:

$$\begin{cases} \delta \dot{x}_1 &= -a\,\delta x_1 + \delta x_2 + a\,\delta h_s \\ \delta \dot{x}_2 &= -(b - \frac{2g_o}{R_o})\delta x_1 + (b - \frac{2g_o}{R_o})\delta h_s + \delta A_z\ \delta x_1 + \delta x_2 + a\delta h_s \\ \delta z &= \delta x_1 \end{cases}$$

$$\Rightarrow \begin{bmatrix} \delta\dot{x}_1 \\ \delta\dot{x}_2 \end{bmatrix} = \begin{bmatrix} -a & 1 \\ -\hat{b} & 0 \end{bmatrix} \begin{bmatrix} \delta x_1 \\ \delta x_2 \end{bmatrix} + \begin{bmatrix} a \\ \hat{b} \end{bmatrix} \delta h_s + \begin{bmatrix} 0 \\ 1 \end{bmatrix} \delta A_z$$

$$\delta z = \begin{bmatrix} 1 & 0 \end{bmatrix} \begin{bmatrix} \delta x_1 \\ \delta x_2 \end{bmatrix} \tag{6.24}$$

Question: How do we implement filter 6.24 with the nonlinear Equations 6.23?

Answer: Replace δ's with full scale signals in Figure 6.11 and $\frac{2g_o}{R_o}$ with $\mu\frac{z+R_o}{||R||^3}$.

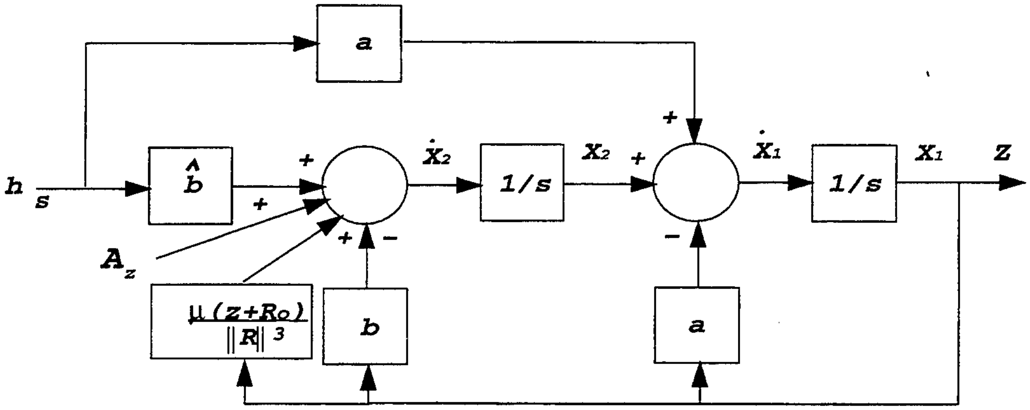


Figure 6.13: Linearized Vertical Channel Filter

Linearize Figure 6.13 at x_o, y_o, z_o, A_{zo} to get Equation 6.24.

$$\begin{cases} \dot{x}_1 &= -ax_1 + x_2 + ah_s \\ \dot{x}_2 &= -bx_1 + \hat{b}h_s + \frac{\mu(x_1+R_o)}{||R||^3} + A_z \\ \dot{z} &= x_1 \end{cases} \tag{6.25}$$

Let

$$x_1 = x_{10} + \delta x_1,\ x_2 = x_{20} + \delta x_2,\ h_s = h_{so} + \delta h_s, \ldots$$

$$\begin{cases} \delta\dot{x}_1 &= -a\delta x_1 + \delta x_2 + a\delta h_s \\ \delta\dot{x}_2 &= -b\delta x_1 + b\delta h_s + \frac{2g_o}{R_o}\delta x_1 + \delta A_z \end{cases} \tag{6.26}$$

Note Equation 6.26 is the same as Equation 6.24.

Summary: Filter design process

1. Obtain INS mechanization equations.
2. Linearize around true values of INS states and inputs.
3. Design a filter for the linear equations obtained in step 2.
4. Implement the filter on the non-linear equations in step 1.

Probability and Stochastic Processes

The errors in accelerometers, altimeters, GPS, gyros, ... are usually modeled as random signals having certain statistical (stochastic properties) properties. This is shown in Figure 6.14.

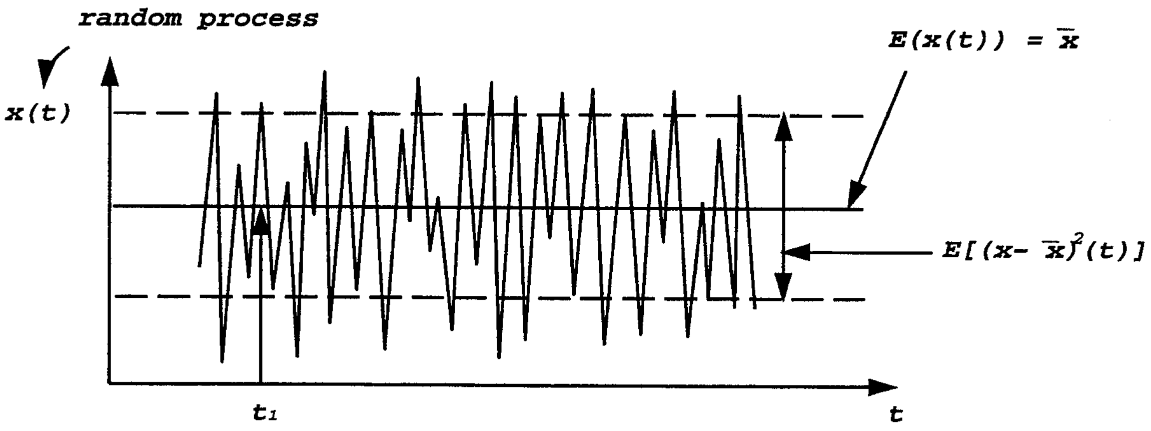


Figure 6.14: Random Process

Definitions

- Freeze time: $x(t_i)$ is a random number with values in some interval of R (real line).
- $E(x(t))$ = average (mean) value of x over time.
- $E[(x(t) - \bar{x}(t))^2]$ = variance of $x(t)$ (rms^2).

- $E[(x(t) - \bar{x}(t))(x(\tau) - \bar{x}(\tau))]$, $t \neq \tau$ is called covariance of $x(t)$ and is denoted by $P_x(t, \tau)$. Note: variance of $x(t) = P(t, t)$.

- Given $x(t)$ and $y(t)$: two random processes, we define

$$P_{xy} = E[(x(t) - \bar{x}(t))(y(\tau) - \bar{y}(\tau))]$$

as the cross–covariance of $x(t)$ and $y(t)$.

If $x(t)$ is a vector of random processes:

$$x(t) = \begin{bmatrix} x_1(t) \\ \vdots \\ x_n(t) \end{bmatrix}$$

$$E(x(t)) = \bar{x}(t) = \begin{bmatrix} E(x_1(t)) \\ \vdots \\ E(x_n(t)) \end{bmatrix}$$

$$\begin{aligned} P_x(t,\tau) &= E[(x(t) - \bar{x}(t))(x(\tau) - \bar{x}(\tau))^T] \leftarrow \text{ is a matrix} \\ &= E\left(\begin{bmatrix} x_1(t) - \bar{x}_1(t) \\ \vdots \\ x_n(t) - x_n(t) \end{bmatrix} \begin{bmatrix} x_1(\tau) - \bar{x}_1(\tau) & \cdots & x_n(\tau) - \bar{x}_n(\tau) \end{bmatrix}\right) \\ &= \begin{bmatrix} E[(x_1(t) - \bar{x}_1(t))(x_1(\tau) - \bar{x}_1(\tau))] & \cdots & E[(x_1(t) - \bar{x}_1(t))(x_n(\tau) - \bar{x}_n(\tau))] \\ \vdots & \ddots & \vdots \\ E[(x_n(t) - \bar{x}_n(t))(x_1(\tau) - \bar{x}_1(\tau))] & \cdots & E[(x_n(t) - \bar{x}_n(t))(x_n(\tau) - \bar{x}_n(\tau))] \end{bmatrix} \end{aligned}$$

P is a $n \times n$ symmetric positive definite matrix with variances on the diagonal and cross–covariances off the diagonal.

<u>Definition</u>: $x(t)$ is called wide sense stationary (wws) if $\bar{x}(t) =$ constant and

$$\begin{aligned} P(t,\tau) &= E[(x(t) - \bar{x}(t))(x(\tau) - \bar{x}(\tau))^T] \\ &= P(t - \tau) \end{aligned}$$

From now on we will deal with $E(\cdot)$ which is a linear operator:

$$\begin{aligned}
E(x_1+x_2) &= E(x_2+x_1)\\
E(\text{const}) &= \text{const}\\
E(Mx) &= ME(x), \text{ where } M \text{ is any constant: scalar, matrix}\\
E(xM) &= E(x)\cdot M\\
\Rightarrow E[(x(t)-\bar{x})(x(\tau)-\bar{x})^T] &= E[x(t)x(\tau)-x(t)\bar{x}^T-\bar{x}x(\tau)^T+\bar{x}\bar{x}^T)]\\
&= E[x(t)x(\tau)-E(x(t))\bar{x}^T-\bar{x}E(x(\tau))^T+\bar{x}\bar{x}^T)]\\
&= E(x(t)x(\tau))-\bar{x}\bar{x}^T-\bar{x}\bar{x}^T+\bar{x}\bar{x}^T\\
&= E(x(t)x(\tau))-\bar{x}\bar{x}^T
\end{aligned}$$

If $y=Mx$, x & : y are *wss* processes and $M\in R^{n\times n}$ – constant matrix.

Then

$$\begin{cases} \bar{y} = E(Mx)=ME(x)=M\bar{x}\\ R_y = M\,R_x\,M^T,\ R_x = \text{covariance of x}\end{cases}$$

Those are important identities. Suppose $x(t_1)$ is independent of $x(t_2), x(t_3), \cdots, x(t_n)$ for all $t_1,\cdots,t_n$.

Then

$$\begin{aligned}
E[(x(t_1)-\bar{x})(x(t_2)-\bar{x})^T] &= E[x(t_1)x(t_2)]-\bar{x}\bar{x}^T\\
&= E(x(t_1))E(x(t_2))-\bar{x}\bar{x}^T\\
&= 0,\ \forall t_1,t_2
\end{aligned}$$

Note, for independent random *wss* process $x(t)$,

$$\begin{aligned}
R_x(t,\tau) &= \begin{bmatrix} E[(x_1(t)-\bar{x}_1(t))(x_1(\tau)-\bar{x}_1(\tau))] & \cdots & E[(x_1(t)-\bar{x}_1(t))(x_n(\tau)-\bar{x}_n(\tau))]\\ \vdots & \ddots & \vdots\\ E[(x_n(t)-\bar{x}_n(t))(x_1(\tau)-\bar{x}_1(\tau))] & \cdots & E[(x_n(t)-\bar{x}_n(t))(x_n(\tau)-\bar{x}_n(\tau))]\end{bmatrix}\\
&= \begin{cases} R,\ t=\tau\\ 0,\ t\neq 0\end{cases}\\
&= R\,\delta(t-\tau), \text{ is symmetric, positive semi–definite matrix}
\end{aligned}$$

Definition: $x(t)$ is white gaussian noise if it is independent and for each fixed time t_i, $x(t_i)$ has gaussian distribution $f(x(t_i))$, Figure 6.15.

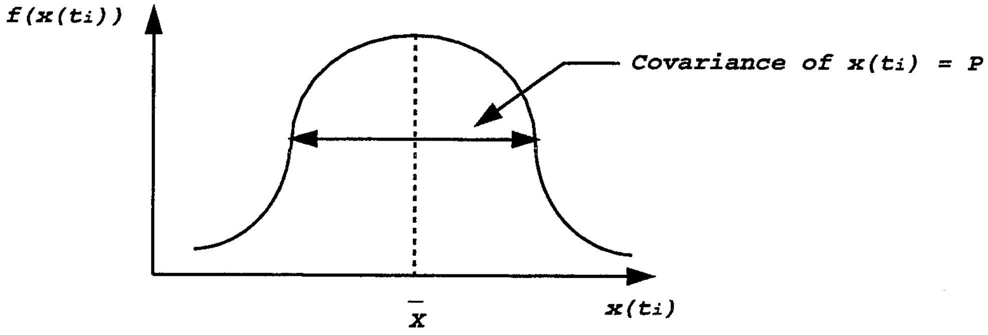


Figure 6.15: Gaussian Distribution

Here x is a scalar random process.

$$f(x(t_i)) = \frac{1}{\sqrt{2\pi R}} \exp\left(-\frac{1}{2}\frac{(x(t_i)-\bar{x})^2}{R}\right)$$

in the matrix case:

$$f(x(t_i)) = \frac{1}{\sqrt{(2\pi)^n det\ P_x}} \exp\left(\frac{1}{2}(x(t_i)-\bar{x})^T P_x^{-1}(x(t_i)-\bar{x})\right)$$

If $x(t)$ is white gaussian we write:

$$x \sim N(\bar{x}, P_x)$$

Finally, consider

$$\dot{x} = Ax + Bw,\ w \sim N(0, R_w)$$

Then

$$\bar{x}(t) = E(x(t)) = e^{A(t-t_o)}E(x(t_o))$$

&

$$P_x(t) = E[(x(t)-\bar{x}(t))(x(t)-\bar{x}(t))^T]$$

is a solution of:

$$\dot{P}_x(t) = A\,P_x(t) + P_x(t)\,A^T + B\,R_w\,B^T$$

If A has all λ's in left half plane, then $P_x(\infty) = P_x$ (steady state covariance):

$$0 = A\,P_x + P_x\,A^T + B\,R_w\,B^T \text{ (Lyapunov Equation)}$$

VII. NAVIGATION USING NAVAIDS

Purpose: Compute vehicle's position and velocity using navaids and determine how accurate these computations are.

Let $x \in R^n$ be the vehicle's position, $(n = 3)$. Let $y \in R^p$ be the vector of navaid measurements $p \leq n = 3$. In general (by geometry, trigonometry, ...) we have

$$y = g(x) \tag{7.1}$$

Let x_o, y_o be nominal quantities of x & y, then

$$x = x_o + \delta x,\ y = y_o + \delta y$$

&

$$\delta y = \left.\frac{\partial g^T}{\partial x}\right|_0 \delta x \tag{7.2}$$

Equations 7.1 and 7.2 represent p equations with n unknowns.

Ex.: Planar navigation using bearing fixes, Figure 7.1.

$$\left\{ \begin{array}{lcl} y_1 & = & \arctan\left[\frac{x_2-a_2}{x_1-a_1}\right] \\ y_2 & = & \arctan\left[\frac{x_2-b_2}{x_1-b_1}\right] \end{array} \right. \Rightarrow y = g(x) \tag{7.3}$$

Measure y_1, y_2: assume (a_1, a_2), (b_1, b_2) are known and solve for x_1, x_2. In Equation 7.3, y_i is called a fix. The nature of the surface $y_i =$ const determines the nature of the fix: linear, conic, circular, hyperbolic, etc... It is difficult and time consuming to solve $y = g(x)$ directly. We need to come up with an interative method:

Let $p = n$ and

$$\begin{array}{lcl} y & = & y_o + \delta y \\ x & = & x_o + \delta x \end{array}$$

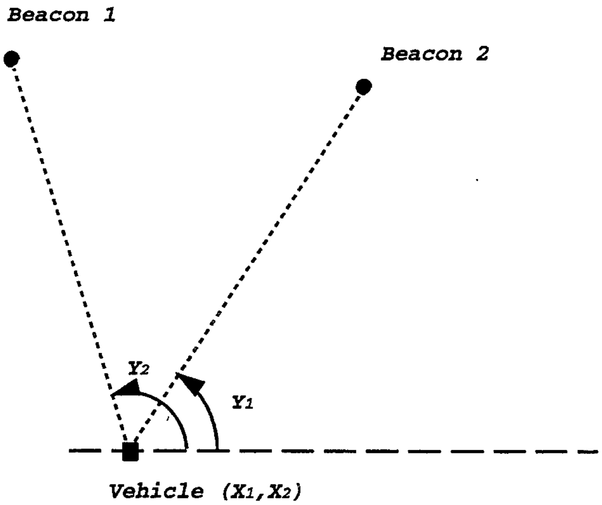


Figure 7.1: Two Bearing Fix

$$\Rightarrow \delta y = \left(\frac{\partial g}{\partial x}\right)\bigg|_0 \delta x \triangleq C\delta x$$

$$\Rightarrow \hat{\delta x} = C^{-1}\delta y \ \text{(if inverse exists)}$$

The estimation mechanization is shown in Figure 7.2.

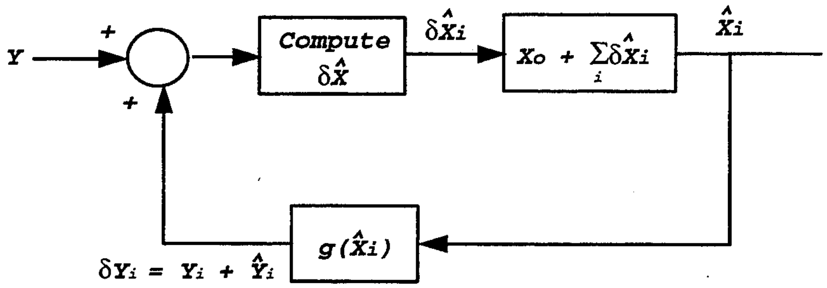


Figure 7.2: Error Estimation

$$\left\{\begin{array}{rcl} \hat{\delta x}_i & = & \left(\frac{\partial g}{\partial x}\right)^{-T}\bigg|_{\hat{x}_i} \delta y_i = C^{-1}\delta y_i \\ \hat{x}_i & = & \hat{x}_{i-1} + \delta\hat{x}_i \\ \delta y & = & y_i - g(\hat{x}_i) \end{array}\right. \qquad \boxed{\text{Estimation Newton Algorithm}}$$

Error Analysis

Usually y is corrupted by noise

$$y = g(x) + v \leftarrow \text{ vector of measurement noise,}$$

Assume v is small and $v_o = 0$: $v = \delta v$

Then

$$\delta y = \left.\left(\frac{\partial g}{\partial x}\right)^T\right|_0 \delta x + v = C\, \delta x + v$$

and the estimate error e_x is:

$$\begin{aligned}
e_x &= \delta x - \hat{\delta x} = \delta x - C^{-1}\,\delta y \\
&= \delta x - C^{-1}(C\,\delta x + v) \\
&= \delta x - \delta x - C^{-1}\,v \\
\Rightarrow e_x &= -C^{-1}\,v = -\left(\frac{\partial g}{\partial x}\right)^{-T} v
\end{aligned}$$

Let $v \sim N(0, R_v)$. Then

$$E(e_x) = E\left(\left.\left(\frac{\partial g}{\partial x}\right)^{-T}\right|_0 v\right) = 0$$

and

$$R_{ex} = C^{-1}\,R_v\,C^{-T} = (C^T\,R_v^{-1}\,C)^{-1} \tag{7.4}$$

This formula allows us to determine the best combination of fixes: "R_{ex} should not be much greater than R_v."

Rule of thumb: Have the fix surfaces as orthogonal as possible, Figure 7.3. ⇒ Columns of C will become orthonormal. The less orthogonal the more dependent the columns of C are, the greater the noise amplification in 7.4.

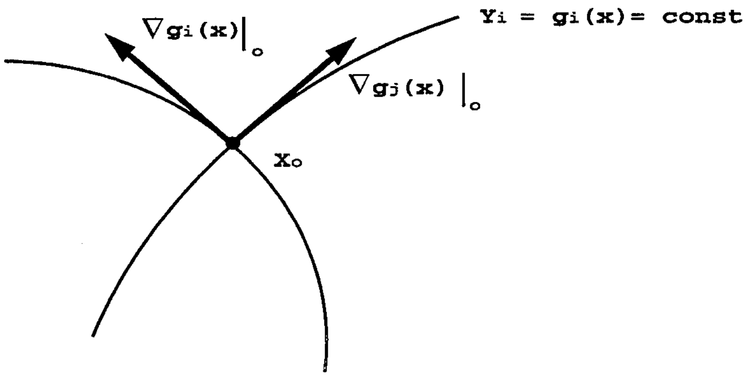


Figure 7.3: Orthogonal Fixes

Note

$$C = \left.\frac{\partial g}{\partial x}^T\right|_0 = \begin{bmatrix} \nabla^T g_1(x)\big|_0 \\ \vdots \\ \nabla^T g_p(x)\big|_0 \end{bmatrix}$$

If C has linearly dependent rows it is not invertible, Figure 7.4:

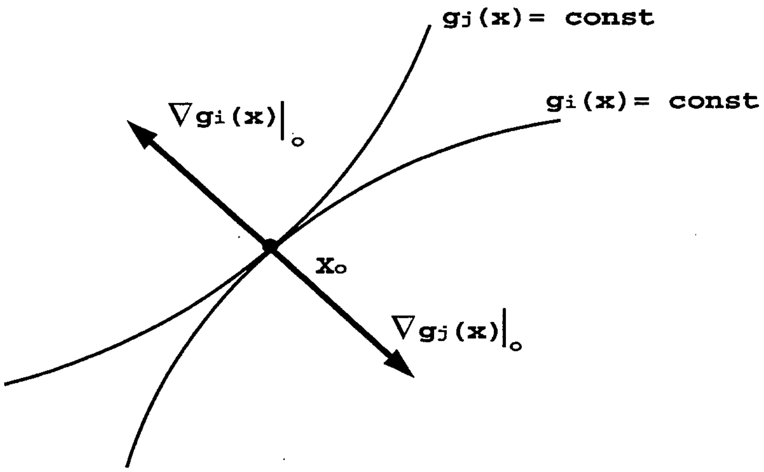


Figure 7.4: Dependent Fixes

In this case we cannot compute x given y.

Example

$$\begin{cases} y_1 &= \tan^{-1}\left[\frac{x_2-a_2}{x_1-a_1}\right] \\ y_2 &= \tan^{-1}\left[\frac{x_2-b_2}{x_1-b_1}\right] \end{cases}$$

Note:

$$\frac{d}{du}\tan^{-1} u = \frac{1}{1+u^2}$$

$$\begin{aligned}
\Rightarrow \frac{\partial y_1}{\partial x_1} &= -\frac{x_2 - a_2}{(x_1-a_1)^2 + (x_2-a_2)^2} \\
\frac{\partial y_1}{\partial x_2} &= -\frac{x_1 - a_1}{(x_1-a_1)^2 + (x_2-a_2)^2} \\
\frac{\partial y_2}{\partial x_1} &= -\frac{x_2 - b_2}{(x_1-b_1)^2 + (x_2-b_2)^2} \\
\frac{\partial y_2}{\partial x_2} &= -\frac{x_1 - b_1}{(x_1-b_1)^2 + (x_2-b_2)^2}
\end{aligned}$$

$$C = \left(\frac{\partial g}{\partial x}\right)^T = \begin{bmatrix} \frac{\partial y_1}{\partial x_1} & \frac{\partial y_1}{\partial x_2} \\ \frac{\partial y_2}{\partial x_1} & \frac{\partial y_2}{\partial x_2} \end{bmatrix}$$

$$e_x = C^{-1} v$$

$$R_{ex} = C^{-1}\, R_v\, C^{-T} = (C^T\, R_v^{-1}\, C)^{-1}$$

Suppose $p > n \Rightarrow$ redundant measurements.

Given

$$y = g(x) + v \tag{7.5}$$

Question: Do redundant measurements help?

First, how can we use redundant measurements? Suppose we know $f_V(v) \equiv$ probability density function of v. Note, the values of v for which $f_V(v)$ is large are more likely to occur than those for which $f_V(v)$ is small:

$$f_V(v = a)dv \simeq P[a \le v \le a + dv] \;\; (dv \text{ is small})$$

From Equation 7.5

$$v = y - g(x)$$

$\Rightarrow$ find x which makes $f_V(v) = f_V(y - g(x))$ large:

$$max_x L(x) = max_x f_V(y - g(x))$$

Here $L(x)$ is called the likelihood function ⇒ maximum likelihood method.

In practice we assume

$$v \sim N(0, R_v)$$

$$\Rightarrow f_V(v) = \frac{1}{\sqrt{(2\pi)^p \, det \, R_v}} \exp\left[-\frac{1}{2}\, v^T \, R_v^{-1} \, v\right]$$

It turns out we only need to maximize

$$L_1(x) = \frac{1}{2}\left[(y - g(x))^T \, R_v^{-1} \, (y - g(x))\right]$$

How:

$$\frac{\partial L_1(x)}{\partial x} = 0 : \; n \text{ equations and } n \text{ unknowns}$$

Recall, in the neighborhood of nominal position

$$\delta y = \left(\frac{\partial g}{\partial x}\right)^T \Bigg|_0 \delta x = C \, \delta x$$

Consider

$$L_1(\delta x) = \frac{1}{2}(\delta y - C \, \delta x)^T \, R_v^{-1} \, (\delta y - C \, \delta x)$$

$$\begin{aligned}
\Rightarrow \frac{\partial L_1(\delta x)}{\partial \delta x} &= \frac{\partial}{\partial \delta x}\left[\frac{1}{2}(\delta y^T \, R_v^{-1} \delta y - \delta x^T \, C \, R_v^{-1} \, \delta y - \delta y^T \, R_v^{-1} \, C \, \delta x + \delta x^T C^T R_v^{-1} C \delta x)\right] \\
&= \frac{1}{2}(-2 C R_v^{-1} \delta y + C^T R_v^{-1} C \delta x) = 0 \\
\Rightarrow \hat{\delta x} &= \left[C^T R_v^{-1} C\right]^{-1} C^T R_v^{-1} \delta y \qquad (7.6)
\end{aligned}$$

Compare equation 7.6 with equation (6.47) in the text. For $R_v = I$ this is the so called "least squares solution".

Example: 1-D navigation. Suppose we measure position x with two errors.

$$\begin{cases} y_1 &= x + v_1 \\ y_2 &= x + v_2 \end{cases}$$

$$\Rightarrow y = \begin{bmatrix} 1 \\ 1 \end{bmatrix} x + \begin{bmatrix} v_1 \\ v_2 \end{bmatrix}$$

Assume

$$v = \begin{bmatrix} v_1 \\ v_2 \end{bmatrix} \sim N\left(\begin{bmatrix} 0 \\ 0 \end{bmatrix}, \begin{bmatrix} 1 & 0 \\ 0 & 3 \end{bmatrix}\right)$$

i.e. y_2 is 3 times less reliable than y_1.

$$\begin{aligned} max_x L_1(x) &= \frac{1}{2}\begin{bmatrix} y_1 - x \\ y_2 - x \end{bmatrix}^T \begin{bmatrix} 1 & 0 \\ 0 & 3 \end{bmatrix}^{-1} \begin{bmatrix} y_1 - x \\ y_2 - x \end{bmatrix} \\ &= \frac{1}{2}\left[(y_1 - x)^2 + \frac{1}{3}(y_2 - x)^2\right] \\ \frac{\partial L_1(x)}{\partial x} &= -(y_1 - x) - \frac{1}{3}(y_2 - x) = 0 \\ \hat{x} &= \frac{3y_1 + y_2}{4} \Rightarrow \text{ weighted sum} \end{aligned}$$

Where more reliable measurements are given more weight.

Note

$$\begin{aligned} \hat{x} &= \left[C^T R_v^{-1} C\right]^{-1} C^T R_v^{-1} y \\ &= \left(\begin{bmatrix} 1 & 1 \end{bmatrix}\begin{bmatrix} 1 & 0 \\ 0 & \frac{1}{3} \end{bmatrix}\begin{bmatrix} 1 \\ 1 \end{bmatrix}\right)^{-1} \begin{bmatrix} 1 & 1 \end{bmatrix}\begin{bmatrix} 1 & 0 \\ 0 & \frac{1}{3} \end{bmatrix}\begin{bmatrix} y_1 \\ y_2 \end{bmatrix} \\ &= \frac{3}{4}\left(y_1 - \frac{1}{3}y_2\right) = \frac{3y_1 + y_2}{4} \end{aligned}$$

Error Analysis $(p > n)$

$$\begin{aligned} e_x &= \delta x - \hat{\delta x} \\ &= \delta x - \left[C^T R_v^{-1} C\right]^{-1} C^T R_v \delta y \\ &= \delta x - \left[C^T R_v^{-1} C\right]^{-1} C^T R_v \left(C\delta x + v\right) \\ &= -\left[C^T R_v^{-1} C\right]^{-1} C^T R_v^{-1} v \\ e_x &= \left[C^T R_v^{-1} C\right]^{-1} C^T R_v^{-1} v \triangleq Mv \end{aligned}$$

$$\Rightarrow R_{ex} \;=\; MR_vM^T = \left[C^TR_v^{-1}C\right]^{-1} C^TR_v^{-1}R_vR_v^{-1}C\left[C^TR_v^{-1}C\right]^{-1} = \left[C^TR_v^{-1}C\right]^{-1}$$

<u>Example</u>

$$\begin{aligned} R_v &= \begin{bmatrix} 1 & 0 \\ 0 & 3 \end{bmatrix};\; C = \begin{bmatrix} 1 \\ 1 \end{bmatrix}; \\ M &= \begin{bmatrix} \frac{3}{4} & \frac{1}{4} \end{bmatrix} \\ R_{ex} &= \begin{bmatrix} \frac{3}{4} & \frac{1}{4} \end{bmatrix} \begin{bmatrix} 1 & 0 \\ 0 & 3 \end{bmatrix} \begin{bmatrix} \frac{3}{4} \\ \frac{1}{4} \end{bmatrix} \\ &= \frac{9}{16} + \frac{3}{16} = \frac{3}{4} \end{aligned}$$

Compare this with noise for $y_1 :\ R_{v1} = 1$ & $y_2 :\ R_{v2} = 3$.

We do better by using both signals.

<u>Use Maximum Likelihood</u>: The mechanization is shown in Figure 7.5.

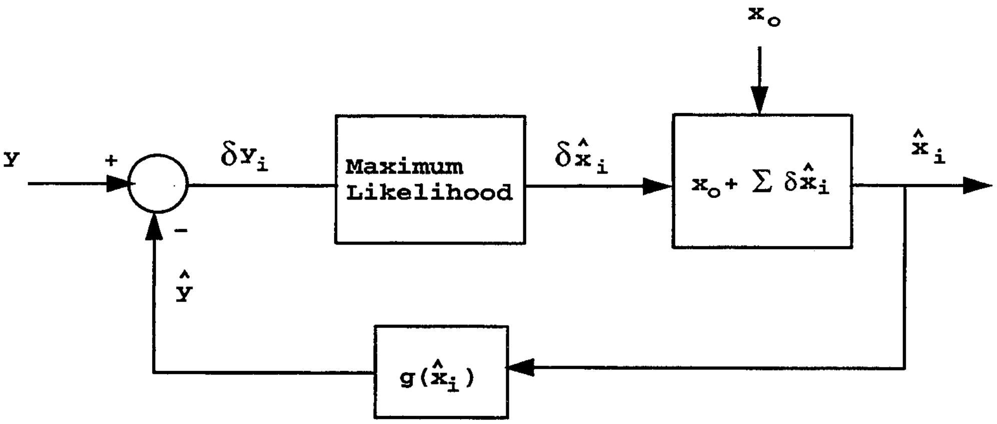


Figure 7.5: Maximum Likelihood Mechanization

$$\begin{cases} C &= \left.\left(\frac{\partial g}{\partial x}\right)^T\right|_{\hat{x}_i} :\ \delta y_i = y_i - \hat{y}_i; \\ \delta\hat{x}_i &= \left[C^TR_v^{-1}C\right]^{-1} C^TR_v^{-1}\delta y_i; \\ \hat{x}_i &= \hat{x}_i^{-1} + \delta\hat{x}_i \\ \delta y_i &= g(\hat{x}_i) \end{cases}$$

Error Model for Wander Azimuth Navigation

$${}^eR = \text{ vehicle position in } \{e\}$$

$$ {}^{e}V = \left(\frac{dR}{dt}\right)_e = V $$

Then

$$ \left\{ \begin{array}{rcl} \left(\frac{d^e R}{dt}\right)_i & = & {}^{e}V + {}^{e}\Omega_{ei} \times {}^{e}R \\ & = & {}^{e}_{p}C\,{}^{p}V + {}^{e}\Omega_{ei} \times {}^{e}R \\ \left(\frac{d^p V}{dt}\right)_p & = & {}^{p}A + {}^{p}g({}^{e}R) - ({}^{p}\Omega_{pi} + {}^{p}\Omega_{ei}) \times {}^{p}V \\ & = & -({}^{p}\Omega_{pe} + 2\,{}^{p}\Omega_{ei}) \times {}^{p}V + {}^{p}A + {}^{p}g({}^{e}R) \end{array} \right. \qquad (7.7) $$

Now, let

$$ \begin{array}{rcl} {}^{p}V & = & {}^{p}V_o + {}^{p}\delta V = {}^{p}V_o + \delta V \\ {}^{e}R & = & {}^{p}R_o + {}^{p}\delta R = {}^{e}R_o + \delta R \\ {}^{p}A & = & {}^{p}A_o + {}^{p}\delta A = {}^{p}A_o + \delta A \end{array} $$

Note, given a, b, c, $\in R^3$ and $c = a \times b$, then

$$ \frac{\partial c}{\partial a} = \frac{\partial(a \times b)}{a} = \frac{\partial(-b \times a)}{\partial a} = \frac{\partial S(-b)a}{\partial a} = S(-b) $$

and

$$ \frac{\partial c}{\partial b} = \frac{\partial(a \times b)}{\partial b} = \frac{\partial S(a)b}{\partial b} = S(a) $$

Therefore, we get

$$ \begin{array}{rcl} \dot{\partial R} & = & {}^{p}_{e}C\delta V + S({}^{e}\Omega_{ei})\,\delta R \\ \dot{\partial V} & = & -S({}^{p}\Omega_{pe} + 2\,{}^{p}\Omega_{ei})\,\delta V + \delta A + \frac{\partial\,{}^{p}g({}^{e}R)}{\partial\,{}^{e}R}\delta R \end{array} \qquad (7.8) $$

Note, in equation 7.8 $(\dot{\ })$ was used for differentiation. This is abuse of notation. However, since we integrating both sides which are resolved in correct coordinate systems, it is OK.

Recall,

$$ {}^{p}g({}^{e}R) = -\frac{\mu}{R^2}\,{}^{p}i_R = -\frac{\mu}{R^2}\,{}^{p}_{e}C\,{}^{e}i_R, $$

The vector i_R is shown in Figure 7.6.

where

$$ {}^{e}i_R = \frac{{}^{e}R}{R} \ \& \ R = \|{}^{e}R\| $$

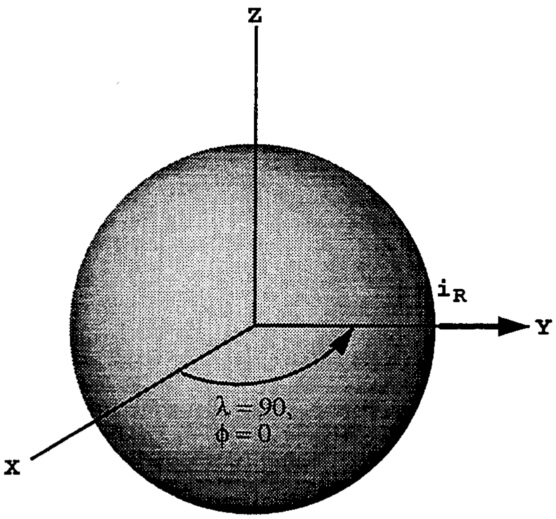


Figure 7.6: Unit Vector, i_R

$$
{}^{e}g = \begin{bmatrix} 0 \\ -\frac{\mu}{R^3} \\ 0 \end{bmatrix}
$$

$$
{}^{p}g = \begin{bmatrix} 0 & 1 & 0 \\ -1 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 0 \\ -\frac{\mu}{R^3} \\ 0 \end{bmatrix} = \begin{bmatrix} -\frac{\mu}{R^3} \\ 0 \\ 0 \end{bmatrix}
$$

Therefore, we get

$$
\begin{aligned}
\frac{\partial\, {}^{p}g(\,{}^{e}R)}{\partial\, {}^{e}R} &= \frac{\partial\, {}^{p}_{e}C\, {}^{p}g(\,{}^{e}R)}{\partial\, {}^{e}R} \\
&= {}^{p}_{e}C\, \frac{\partial\, {}^{p}g(\,{}^{e}R)}{\partial\, {}^{e}R} \\
&= {}^{p}_{e}C \begin{bmatrix} -\frac{g_o}{R_o} & 0 & 0 \\ 0 & -\frac{g_o}{R_o} & 0 \\ 0 & 0 & -\frac{g_o}{R_o} \end{bmatrix}
\end{aligned}
$$

Finally, we obtain

$$
\frac{d}{dt} \begin{bmatrix} \delta R \\ \delta V \end{bmatrix} = \begin{bmatrix} S\,(\,{}^{e}\Omega_{ei}) & {}^{p}_{e}C \\ {}^{p}_{e}C \frac{g_o}{R_o} & -S\,(\,{}^{p}\Omega_{pe} + 2\,{}^{p}\Omega_{ei}) \end{bmatrix} \begin{bmatrix} \delta R \\ \delta V \end{bmatrix} + \begin{bmatrix} I \\ 0 \end{bmatrix} \delta A
$$

We should be able to verify this model in *Simulink*.

VIII. THE NAVSTAR GPS

The NAVigation Satellite Timing And Ranging Global Positioning System is a satellite–based radio navigation system with the capability to provide locating data to an unlimited number of users [1]. The first satellite was deployed in 1978 although the first receiver did not become commercially available until 1982 [Ref. 10, section 1.2]. This system is the product of experience gained from several previous space–based navigation systems like TRANSIT AND USAF System 621B. It is comprised of three segments:

- Space segment
- User segment
- Control segment

Each segment is discussed in the following sections.

A. SPACE SEGMENT

A total of 24 satellites now constitute a fully operational space segment. Twenty–one of theses space vehicles (shown in Figure 8.1 operate continuously, while the remaining three act as orbiting spares. Today, only three Block I satellites remain in orbit. These satellites were the first GPS satellites in space. They were launched from 1978 through 1985 [Ref. 10, section 1.2]. Block I space vehicles weigh 960 pounds and generate 420 watts of electrical power [Ref. 8, p. 27]. The remaining satellites in orbit were creatively named the Block II vehicles. These vehicles are less vulnerable to radiation and have more memory. With this increase in capability has come an

[1] This chapter is an abridgement of Marquis [Ref. 5, Chap. 2] and is included for continuity.

increase in weight and power requirements. These newer satellites weigh 2000 pounds and generate 700 watts of electrical power.

The space vehicles are inserted into orbits defined by the six Keplerian constants:

- semi–major axis, a
- orbital eccentricity, e
- orbital inclination, i
- ascending node, Ω
- argument of perigee, ω
- time of perigee passage, T

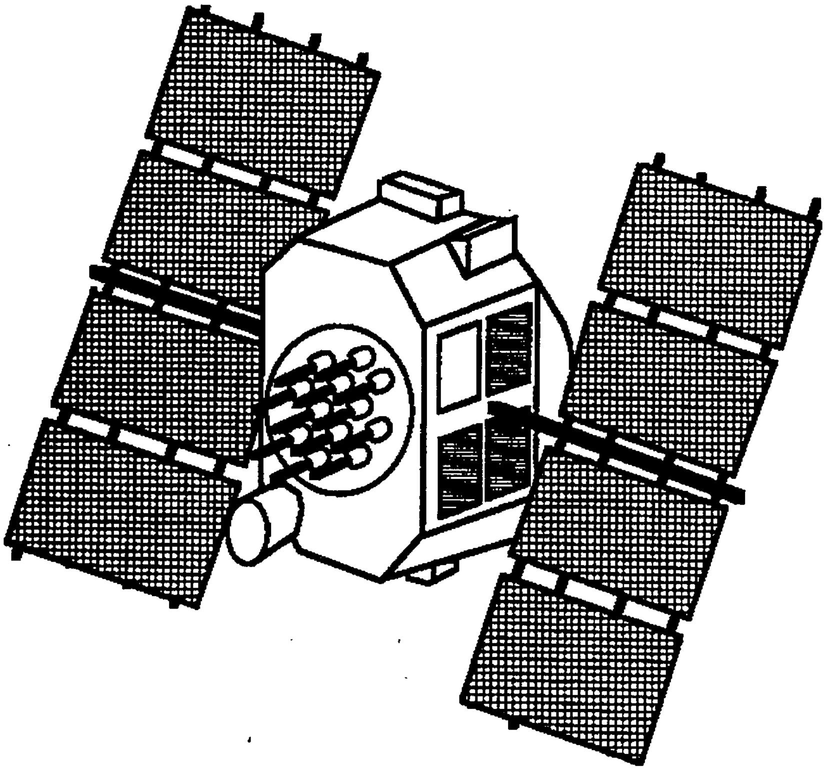


Figure 8.1: A NAVSTAR GPS Satellite from [Ref. 5, p. 4.01]

The semi–major axis of a GPS satellite orbit is nominally 26,560 kilometers. It is half of the length of the ellipse which defines the space vehicle's path. Second, the eccentricity (different from the eccentricity defined with respect to the shape of the WGS–84 ellipsoid) or oblateness of the satellite's orbit is defined as follows:

$$e = \frac{r_a - r_p}{r_a + r_p}, \tag{8.1}$$

where r_a is the apogee radius and r_p is the perigee radius. For GPS satellites, the eccentricity cannot exceed two percent. The third of the Keplerian elements is orbital inclination. It is the angle between the plane defined by the orbit and the equator. For example, satellites in a polar orbit have a 90° orbital inclination; those in an equatorial orbit have 0° orbital inclination. Block I NAVSTAR GPS vehicle orbits are inclined at 63° while Block II satellites are inclined at 55°. The ascending node is the satellite's geodetic longitude as it passes through 0° of latitude toward the northern hemisphere. GPS satellites orbit in six different planes. Thus, there are exactly six different ascending nodes. The last two Keplerian elements are the argument of the perigee and the time of perigee passage. The argument of the perigee is the angle in the orbital plane between the ascending node and the closest point of approach of the satellite to earth. The time when the vehicle reaches this point is the time of perigee passage. The ranges of these last two parameters span all possible values for GPS satellites. That is, satellites reach their perigee at all different times of day and different locations.

Ideally, the six Keplerian elements would be sufficient to define any satellite's three–dimensional position and velocity vectors for all time. However, the orbits become perturbed by lunar and solar gravity and the earth's equatorial bulge, as well as several other less significant effects. Thus, the number of quantities required to fully specify the position and velocity of a satellite in a real orbit is increased to 16. The extra elements consist of time rates of change of the ascending node and inclination, *i. e.* $\dot{\Omega}$ and $\dot{i}$, as well as six other coefficients which account for variation in the earth's gravitational field. These 16 coefficients must become part of the navigation message.

The navigation message is the framework in which the GPS satellites broadcast their data. One "frame", the complete message, consists of five subframes of 300 bits each. Subframe one contains coefficients used to correct the satellite's clock to exact GPS time. The 16 pieces of ephemeris data are broadcast in the second and third subframes. Subframe four contains special messages, ionospheric correction coefficients, and coefficients for conversion of GPS time to Universal Coordinated Time. Ephemeris and health data for the entire GPS constellation is transmitted in subframe five. Because of the volume of data in subframes four and five, both must be subdivided into 25 pages. Therefore, it takes 25 full frames, broadcast at the rate of 30 seconds per frame, or 750 seconds (12.5 minutes) to receive the entire navigation message. Critical navigation data – ephemerides, and clock correction coefficients — are updated every frame. Secondary data transmitted in subframes four and five is provided primarily to assist the receiver in acquiring other satellites. These data are not intended to be precise so the lower update rate of once every 12.5 minutes is satisfactory.

Satellites use a pseudorandom binary code superimposed on the two carrier frequencies to communicate. The two frequencies are L1 (1575.42 MHz) and L2 (1227.6 MHz). The L1 frequency carries both the C/A– (coarse acquisition) and P–(precise) codes, while L2 carries only the P–code. The less precise C/A–code is broadcast at a rate of 1.023 million bits per second (Mbps) and is 1023 bits long. Therefore, this code repeats itself every millisecond. The C/A–code is unique to each satellite and does not change, allowing GPS receivers to quickly distinguish between space vehicles, even without access to the P–code. The P–code, being more precise, is transmitted at 10.23 Mbps with a code length of approximately six trillion bits. This code takes 37 weeks to repeat. Since the codes are reset every week at midnight between Saturday and Sunday, there are sufficient "code weeks" available in the P–code such that one can be assigned to each space vehicle each week. Therefore, GPS receivers can easily distinguish satellites from each other with their individual, weekly–assigned P–codes. The remaining code weeks are available for uplink from the control segment to the satellites.

B. USER SEGMENT

The many thousands of GPS receivers constitute the user segment. The receiver's functions are to receive and interpret the navigation message and to calculate and output position. GPS receivers must determine the time the navigation message takes to travel the distance from the satellite to the receiver. This is achieved by autocorrelating the pseudorandom binary pulse train received from the satellite with the one in memory. A typical civilian GPS receiver must have the C/A–codes for all 24 satellites in its memory (requiring only three kilobytes). As the pseudorandom code is received, the receiver slews its code until the result of the autocorrelation function jumps to one. The receiver is now "locked–on" to that code. Multiplying the length of time that the receiver must slew its code to achieve a unity correlation by the speed of light yields the "pseudorange". This is not the actual range because the offset of the receiver clock is uncertain. When the receiver can lock–on to four satellites and thus measure pseudoranges to each, is three–dimensional position and clock error can be solved for by inverting this set of four equations

$$
\begin{aligned}
\rho_1 &= \sqrt{(x_{sat_1} - x_{rcvr})^2 + (y_{sat_1} - y_{rcvr})^2 + (z_{sat_1} - z_{rcvr})^2} + c\Delta t \\
\rho_2 &= \sqrt{(x_{sat_2} - x_{rcvr})^2 + (y_{sat_2} - y_{rcvr})^2 + (z_{sat_2} - z_{rcvr})^2} + c\Delta t
\end{aligned}
$$

$$\begin{aligned}\rho_3 &= \sqrt{(x_{sat_3} - x_{rcvr})^2 + (y_{sat_3} - y_{rcvr})^2 + (z_{sat_3} - z_{rcvr})^2} + c\Delta t \\ \rho_4 &= \sqrt{(x_{sat_4} - x_{rcvr})^2 + (y_{sat_4} - y_{rcvr})^2 + (z_{sat_4} - z_{rcvr})^2} + c\Delta t, \qquad (8.2)\end{aligned}$$

where ρ_i are pseudoranges, $[x_{sat_i}, y_{sat_i}, z_{sat_i}]$ are the ECEF coordinates of a satellite, $[x_{rcvr}, y_{rcvr}, z_{rcvr}]$ are the ECEF coordinates of the GPS receiver, c is the speed of light, and Δt is the receiver's clock error. Having solved this set of equations, the receiver now need only transform the solution to the geodetic system and display the results.

C. CONTROL SEGMENT

The GPS Control segment is responsible for generating and uplinking clock correction coefficients and ephemeris corrections for all satellites in the constellation. Five control stations — Hawaii, Ascension Island, Diego Garcia, Kwajalein, and Colorado Springs, Colorado – comprise this segment. These control stations are essentially GPS receivers capable of constantly tracking all satellites in view. Additional capabilities include highly accurate Cesium clocks and recording facilities. The first four of these stations track all satellites in view and record pseudorange information continuously. This data is sent to the Master Control Station at Colorado Springs where it is processed. When all four monitor stations have locked-on to a single space vehicle simultaneously, the exact inverse of the standard navigation problem mentioned in the previous section exists. Since the exact locations of the monitor stations are known and their clocks are extremely accurate, the four unknowns become the three coordinates of the space vehicle position and its clock offset. The Master Control Station calculates these quantities and from them, derives the necessary ephemeris and clock corrections. This information is uplinked to each space vehicle at least daily.

D. DIFFERENTIAL GPS

Although GPS alone provides highly accurate positioning, it can be made still more accurate by augmenting it with a differential station. A differential station is merely another GPS receiver whose exact location is known. When this second receiver is near the first receiver, both are subject to nearly the same errors, *i. e.* the same local atmospheric properties, nearly identical elevation angles and propagation paths to any given GPS satellite, the same clock errors and the same ephemeris errors for each satellite. By employing the Pythagorean theorem on its position and the satellite position broadcast in the navigation message, the differential station can calculate the exact range to that satellite. Meanwhile, it can also calculate pseudorange in the standard way (see Section B.). By comparing these two values for each satellite in view, the differential station can evaluate the pseudorange error to each satellite. These values can be broadcast periodically to be used by receivers in the local area to improve accuracy. By using the differential corrections, GPS accuracy, even for single frequency C/A–code users, can be improved to one to seven meters rms [Ref. 8, p. 64].

E. GPS ERROR SOURCES

Despite its exceptional accuracy, GPS is subject to numerous error sources. Clearly, the major error sources must be included in the DGPS model. Error sources are:

- atmospheric delays
- Selective Availability
- clock differences
- ephemeris error

- multipath

- receiver noise

- Dilution of Precision (DOP)

Each of these error sources is discussed in detail in the following sections.

1. Atmospheric Delays

a. Ionospheric Delays

The ionosphere is a layer of charged particles between 100 and 1000 kilometers above the earth's surface. These particles interact with the transmitted GPS signal and slow it, increasing pseudoranges. The equation describing this delay is

$$\Delta t = \frac{40.3}{cf^2} TEC, \tag{8.3}$$

where Δt is the delay in seconds, c is the speed of light (3×10^8 m/s), f is the system frequency (1575.42 MHz for L1), and TEC is the Total Electron Content (electrons/m^2) along the signal's path. The TEC is strongly effected by the solar cycle, season, time of day, and latitude. TEC maxima occur:

- daily: 1400 local

- seasonally: spring equinox

- solar cycle: every 11 years (next 2001-2002)

- geographic: 20° magnetic latitude

It varies ± 25% rms at all latitudes during daylight [Ref. 10, section 2.5, p. 24]. The pseudorange error due to ionospheric effects can be as great as 40 meters [Ref. 10, section 2.5, p. 13].

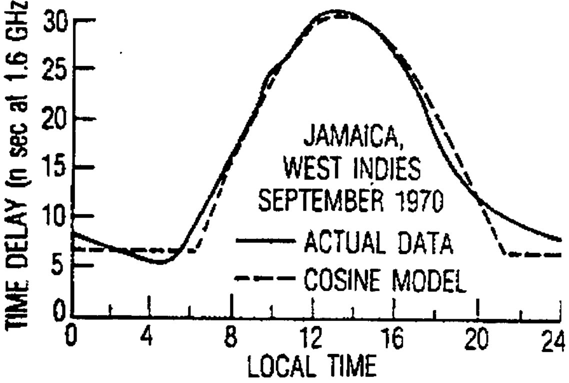


Figure 8.2: Diurnal Ionospheric Delay from [Ref. 6, p.208]

The algorithm described in [Ref. 12] which removes 55–60% of the ionospheric delay is based on Figure 8.2, which shows the typical diurnal variation of the ionospheric delay. The "ACTUAL DATA" curve shown in Figure 8.2 is modeled with the "COSINE MODEL" curve, a half–cosine. The equation of the model curve is

$$\Delta t = DC + A \cos \frac{2\pi(t - T_p)}{P}, \tag{8.4}$$

where DC, A, T_p, and P (constant offset, amplitude, phase, and period, respectively) describe the diurnal variation of the ionospheric delay. DC and T_p are assumed constant at five nanoseconds and 1400 local time, respectively. Amplitude and period are each modeled as four term power series as follows:

$$\begin{aligned} A &= \sum_{n=0}^{3} \alpha_n \phi_m^n \\ P &= \sum_{n=0}^{3} \beta_n \phi_m^n, \end{aligned} \tag{8.5}$$

where the α_n's and β_n's are constants which are broadcast in the GPS navigation message, chosen based on the day of the year and average solar flux over the past five days, and ϕ_m is the geomagnetic latitude of the ionospheric subpoint. The ionospheric subpoint is the intersection of the line between the space vehicle and the receiver with the surface at the mean height of the ionosphere.

Next, one must first find the subtended earth angle between the user and the satellite, EA (degrees)

$$EA = \frac{445}{el + 20} - 4, \tag{8.6}$$

where el is the elevation angle of the satellite with respect to the user in degrees. Knowing EA, the geodetic location of the ionospheric subpoint can be approximated by

$$\begin{aligned} \phi_I &= \phi_{rcvr} + EA \cos az \\ \lambda_I &= \lambda_{rcvr} + \frac{EA \sin az}{\cos \phi_I}, \end{aligned} \tag{8.7}$$

where ϕ and λ denote geodetic latitudes and longitudes, respectively and az is the azimuth of the satellite with respect to the receiver. Now the geodetic latitude can be converted to geomagnetic latitude (the required quantity) with the following approximation:

$$\phi_m = \phi_I + 11.6 \cos(\lambda_I - 291), \tag{8.8}$$

where all angles are in degrees.

The dimensionless scale factor (SF) which scales the entire delay is

$$SF = 1 + 2(\frac{96 - el}{90})^3. \tag{8.9}$$

The final expression for Δt is a three term Taylor series expansion of Equation 8.4 is

$$\Delta t = \begin{cases} SF \cdot [DC + A(1 - \frac{x^2}{2!} + \frac{x^4}{4!})] & \text{for } \mid x \mid < \frac{\pi}{2} \\ SF \cdot [DC] & \text{for } \mid x \mid \geq \frac{\pi}{2} \end{cases}$$

where

$$x = \frac{2\pi(t - T_p)}{P}. \tag{8.10}$$

Due to the 25% rms variation, the error is modeled with a 25% standard deviation. Therefore, the random part of the error remains within ±25% of nominal 68% of the time.

b. Tropospheric Delays

The lower section of the atmosphere also causes signal propagation delays. Typical tropospheric delay is approximately two meters for 90° satellite elevation (directly overhead) up to 28 meters at a five degree elevation angle [Ref. 13, p. 218]. In this application, the atmosphere can be modeled as being composed of "wet air" and "dry air". Dry air is responsible for 90% of the total tropospheric delay, whereas, wet air is responsible for only ten percent. While the moisture content in the troposphere is virtually impossible to model accurately, this inaccuracy has minimal impact. Numerous models which calculate the tropospheric delay have been developed. Black developed the following model in [Ref. 14]. Let

$$\Delta s = \Delta s_d + \Delta s_w,$$

where

$$\begin{aligned}
\Delta s_d &= 2.343P \cdot [\frac{(T-4.12)}{T}] \cdot I(h = h_d, E), \\
\Delta s_w &= k_w \cdot I(h = h_w, E), \\
I(h, E) &= \{1 - [\frac{\cos E}{(1 + (1 - l_c) \cdot \frac{h}{r_s}}]^2\}^{(-1/2)}, \\
h_d &= 148.98(T - 4.12)\,\text{m}, \\
h_w &= 13{,}000\,\text{m}, \\
k_w &= 0.2, \\
r_s &= 6378137\,\text{m}, \\
P &= 1\,\text{atm}, \\
T &= 15\,^\circ\,\text{C},
\end{aligned} \tag{8.11}$$

and Δs is the wet or dry delay in meters, r_s is the distance from the center of the earth to the station, P is the surface pressure in atmospheres, E is the satellite elevation angle, and T is the surface temperature in degrees Celsius. It should be noted that l_c is an empirical constant. The value of 0.85 is only valid for elevation angles above five degree (GPS receivers typically ignore satellites at lesser elevation angles). Similarly, k_w is an empirical constant which varies based on latitude and season. The value 0.20 corresponds to the value for spring or fall in mid–latitudes. This model has been shown to be virtually exact at elevation angles greater than 40°, with its worst error of about 0.045 m occurring between five and ten degrees of elevation.

This tropospheric model is assumed to vary 15% from the nominal value. Therefore, it is modeled with a 7.5% nominal standard deviation. This maintains the random part of the error within 15% of the model value 95% of the time.

2. Selective Availability

Selective Availability is a method that the Department of Defense can use to intentionally degrade the accuracy of pseudorange measurements. Typically, this is accomplished by dithering the space vehicle clock signal. Dithering the clock involves encoding the binary time signal the space vehicle broadcasts. The decryption process is classified and available only to DOD authorized users.

The use of SA essentially results in the satellites' "lying" to the receiver about their position. Clearly, this adversely impacts precision. Currently, SA is in operation on all Block II space vehicles which comprise the majority of the constellation. The DOD's stated goal for the positioning accuracy under SA is 100 meters (twice rms) for a two–dimensional fix [Ref. 15]. According to [Ref. 16, p. 420], the selective availability error can be modeled as a zero-mean, five meter standard deviation low frequency noise. The suggested cutoff frequency for SA noise is $\frac{1}{180}Hz$.

3. Clock Differences

The clock model used in this treatment of DGPS is a two state model shown in Figure 8.3. It is reasonable to expect the clock to have both a bias and drift. From daily exposure to clocks, the average person realizes that most clocks are slightly offset from correct time (bias), and that their accuracy tends to degrade with time (drift). From an engineering standpoint, these two phenomena can be best modeled with zero mean, white, Gaussian noise. Rewriting the model in state space form for further analysis yields

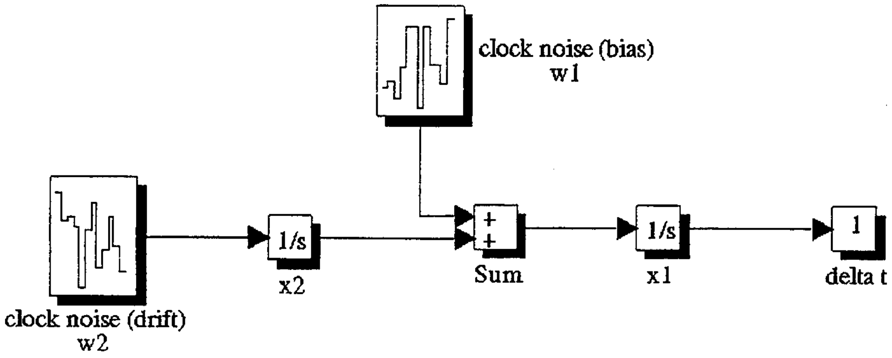


Figure 8.3: Receiver Clock Model

$$\dot{x} = \begin{bmatrix} 0 & 1 \\ 0 & 0 \end{bmatrix} x + \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} w$$

where

$$E(ww^T) = \begin{bmatrix} S_1 & 0 \\ 0 & S_2 \end{bmatrix}. \tag{8.12}$$

The covariance of the clock error state (x_1) can be found by solving the Lyapunov equation which can be found numerous control textbooks, one of which is [Ref. 17, p. 104]. This equation must be solved over a finite time interval (Δt) because the two state clock model is unstable. This interval would normally be the sampling time, if the model were discrete. The result is

$$E(x_1^2) = S_1\Delta t + \frac{S_2\Delta t^3}{3}. \tag{8.13}$$

By taking the square root of the variance and dividing by Δt, one finds the more standard clock parameter the Allan variance, V_A

$$V_A = \sqrt{\frac{S_1}{\Delta t} + \frac{S_2\Delta t}{3}}. \tag{8.14}$$

A representative plot of the two state model Allan variance as a function of averaging time (*i. e.* Δt) is shown in Figure 8.4.

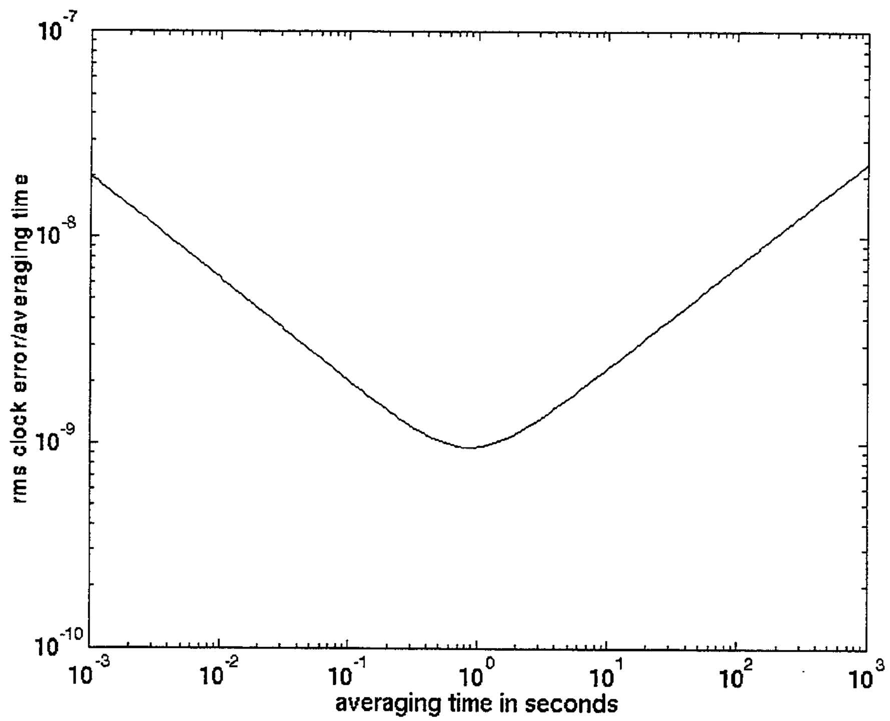


Figure 8.4: Ideal Allan Variance

Real clocks behave somewhat differently from this simple model. A typical Allan variance plot for a real clock is shown in Figure 8.5.

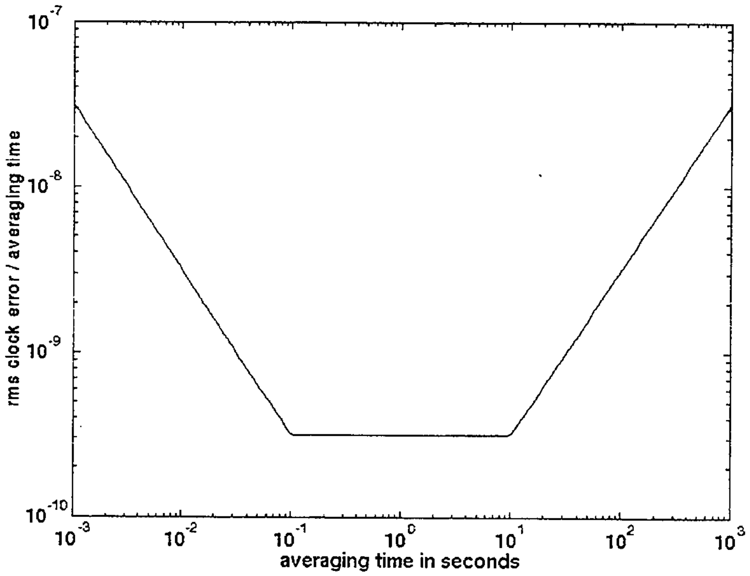


Figure 8.5: Real Allan Variance

The flat portion of the curve is called the flicker floor. It is the result of a non-linear effect which cannot be modeled by the two state model. This causes a significant discrepancy between this simple model and the real world.

In order for the model to better represent reality, it must be carefully crafted to fit the actual plot as much as possible. By carefully choosing the values of S_1 and S_2, it is possible to make the actual and model curves fairly close. The key Allan variance parameters between 0.1 and ten seconds of averaging time are h_0, h_{-1}, and h_{-2} [Ref. 18]. Values of these three parameters for three common GPS timing standards are shown in the Table 8.1 from [Ref. 16, p. 428].

Brown [Ref. 16] finds that one must choose where the 2-state model is accurate. Since normal averaging times are in the 0.1 to ten second interval already mentioned, this is the region where the model is made accurate. Maximizing the accuracy of the model in that region dictates the following values for the noise spectral

TABLE 8.1: ALLAN VARIANCE PARAMETERS FOR THREE COMMON TIMING STANDARDS

timing standard	h_0	h_{-1}	h_{-2}
crystal	2×10^{-19}	7×10^{-21}	2×10^{-20}
ovenized crystal	8×10^{-20}	2×10^{-21}	4×10^{-23}
Rubidium	2×10^{-20}	7×10^{-24}	4×10^{-29}

densities:

$$\begin{aligned} S_1 &\sim \frac{h_0}{2} \\ S_2 &\sim 2\pi^2 h_{-2}\ \mathrm{s}^{-2}. \end{aligned} \tag{8.15}$$

In order to remain conservative in this generic model of DGPS, the least accurate clock — the crystal clock — is used. The values for S_1 and S_2 for this clock are

$$\begin{aligned} S_1 &= 4 \times 10^{-19} \\ S_2 &= 1.58 \times 10^{-18}\ \mathrm{s}^{-2}. \end{aligned} \tag{8.16}$$

These values are be used in the DGPS error model.

4. Ephemeris Error

In converting the pseudoranges of at least four satellites (six in this model) to a three–dimensional position and clock error, one must solve a series of non-linear, coupled algebraic equations. In these equations, the positions of the satellites are critical. The only way a GPS receiver or navigation filter knows the space vehicle positions is the navigation message. If broadcast satellite positions are incorrect, the accuracy of the resulting receiver position suffers. The control segment of the GPS system maintains positions on the entire constellation quite accurately.However, it would be folly to expect the satellites to broadcast inerrantly accurate positions.

Typical ephemeris inaccuracies according to [Ref. 19] are shown in Figure 8.6.

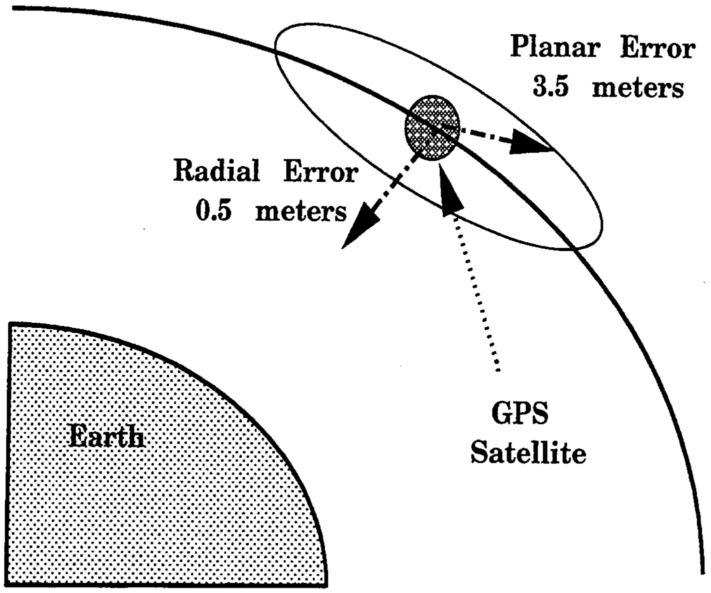


Figure 8.6: Ephemeris Error

These errors are resolved in a coordinate system local to each space vehicle. The three mutually perpendicular directions are radial, along track, and cross track. Because the DGPS model developed in this thesis does not account for satellite motion, along track and cross track directions cannot be resolved. To remain conservative, these two errors are combined into a circular error in the plane they define. This error is modeled as zero mean, white, Gaussian noise with a standard deviation of 3.5 meters. Likewise, the radial error is modeled as zero mean, white Gaussian noise with a 0.5 meter standard deviation. Both of these errors make the stated accuracy the two rms point. In other words, the value stays within the stated accuracy 95% of the time.

Since the space vehicle location enters the model in earth centered, earth fixed Cartesian coordinates while the error is added in geodetic coordinates, a transformation is between the two coordinate systems must be performed. Having converted the satellite positions to geodetic coordinates, the random errors can be added.

However, since the geodetic coordinates contain angles, the random errors in position must be converted to equivalent angles in latitude and longitude by dividing by the radius of the space vehicles' orbits, ~26,560,000 meters. The position and the error are now in compatible coordinates and can thus be summed.

5. Multipath

The signal radiated by a satellite is not required to take a direct path to a receiver. If the signal encounters an electromagnetically reflective object, it may bounce off that object and still find its way to the receiver. This signal has now traveled a greater distance than the straight line joining the space vehicle and the receiver. Because the receiver assumes the direct path is used, this *multipath* phenomenon can introduce pseudorange errors. Due to the satellite/receiver geometry, multipath is far more likely at low elevation angles. The GPS system has several attributes that minimize multipath effects [Ref. 10]:

- The L band frequency (1227.6 MHz) tends to undergo diffuse rather than specular reflection.
- The receiver antennas tend to reject multipath signals.
- The navigation message is broadcast with circular polarization. Circularly polarized signals undergo reversal upon reflection.
- GPS receivers generally use mask angles (the elevation angle below which the satellite is ignored) of at least five degrees.

All of these factors tend to attenuate the strength of any reflected signal making the multipath effect insignificant. Therefore, it is not modeled.

6. Receiver Noise

All receivers corrupt the signals they receive. GPS receivers are no exception. Inaccuracies resulting from quantization error, loop tracking errors and other hardware inadequacies corrupt the pseudorange accuracy. According to [Ref. 10], representative GPS receiver noise has a standard deviation of 7.5 m. Receiver noise is thus modeled as a zero mean, white Gaussian noise with 7.5 m standard deviation. As the differential station and the aircraft receivers are assumed identical, both aircraft pseudoranges and differential station pseudoranges are contaminated with this noise. The noise in the two receivers is assumed to be independent.

7. Dilution of Precision

All of the errors discussed to this point directly effect pseudorange accuracy. The various delays and noise sources can cause the receiver's evaluation of range to the space vechicle's to be inaccurate. However, the pseudoranges themselves are irrelevant. The science of navigation is concerned with *positioning.* Dilution of Precision is the effect that links pseudorange accuracy to position accuracy. DOP can be further classified into several different types:

- VDOP — Vertical DOP (z)
- HDOP — Horizontal DOP (x,y)
- PDOP — Position DOP (x,y,z)
- TDOP — Time DOP (t)
- GDOP — Geometric DOP (x,y,z,t)

The equation which relates DOP and pseudorange error is

$$\sigma_p = \text{DOP} \cdot \sigma_0, \qquad (8.17)$$

where σ_p is the standard deviation of the position error and σ_0 is the standard deviation of the pseudorange error, often called the User Equivalent Range Error (UERE).

Dilution of Precision is a function of satellite to receiver geometry. As Figure 8.7 shows for a four satellite constellation, GDOP is minimized with the space vehicles spread out as much as possible. In fact, the volume of the tetrahedron formed by the unit vectors from the receiver to each satellite is an empirical measure of DOP. PDOP is inversely proportional to the volume of the tetrahedron. If, for example, all of the space vehicles a receiver was using for positioning were in the same plane, PDOP would approach infinity (the volume of the tetrahedron would be zero). Likewise, minimum PDOP is achieved with the geometry shown at the right of Figure 8.7.

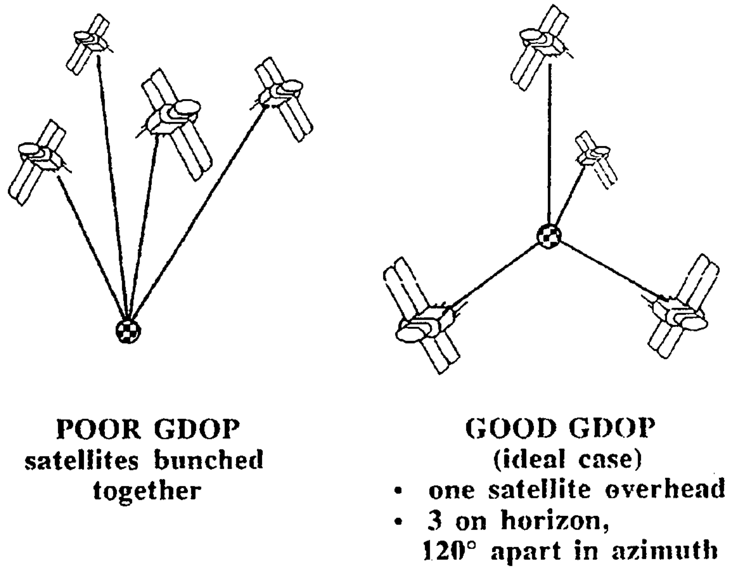

POOR GDOP
satellites bunched together
GOOD GDOP
(ideal case)
• one satellite overhead
• 3 on horizon, 120° apart in azimuth

Figure 8.7: Dilution of Precision from [Ref. 5, p. 4.22]

The goal for the design of the entire GPS constellation of satellites is a PDOP no greater than six everywhere on the earth. With the entire set of 24 space vehicles now in orbit, users can expect PDOP values under three [Ref. 20].

The entire concept of GPS navigation has now been thoroughly discussed. All of the information put forth in this discussion will be used in developing the DGPS computer model in the next chapter. To complete the sensor discussion, a description of INS follows.

F. INERTIAL NAVIGATION

Inertial navigation has long been the standard for self–contained, long–range aircraft navigation. An Inertial Navigation System (INS) senses aircraft thrust acceleration, angular rates and spatial orientation resolved in an orthogonal system and computes the inertial acceleration. This acceleration can now be integrated — once to find velocity, and twice to find position.

The primary component in any inertial navigation system is the Inertial Measuring Unit (IMU). The IMU is composed of three accelerometers, three rate gyros, and two inclinometers. The accelerometers measure thrust acceleration. Thrust acceleration is composed of linear, centripetal, and gravitational effects. Einstein's Principle states that it is impossible for a sensor to distinguish between the effects of gravity and acceleration. Thus, the thrust acceleration that it provides is

$$^{b}a = ^{b}\dot{v} + ^{b}\omega \times ^{b}v + ^{b}g, \tag{8.18}$$

where ^{b}a is the thrust acceleration, $^{b}\dot{v}$ is the linear acceleration, $^{b}\omega$ is the angular velocity, ^{b}v is the velocity, and ^{b}g is gravity. All quantities are in the body frame. This principle necessitates that the computation portion of the INS compute and remove local gravity from the measured accelerations.

Rate gyros sense angular velocities. These angular velocities can be resolved in either the body or inertial frame, depending on the type of IMU implementation. The two inclinometers sense aircraft inertial orientation, *i.e.*, Euler angles.

There are two conceptual methods for implementing inertial navigation:

- Gimbaled IMU
- Strapdown IMU

Brief discussions of each method follow.

1. Gimbaled IMU

A gimbaled IMU rotates about its four gimbals during operation (see Figure 8.8). Aircraft IMU's must have four gimbals to prevent gimbal lock, while earthbound IMU's require only three gimbals [Ref. 21, p. 192]. A controller maintains the IMU in a constant inertial orientation toward true North and lying in the locally horizontal plane. By maintaining this orientation, the gimbaled IMU measures inertial quantities directly. This data can be integrated without transformation to yield inertial velocity, position and Euler angles. From a navigation standpoint, this configuration seems ideal. However, gimbaled systems are large and heavy, making them

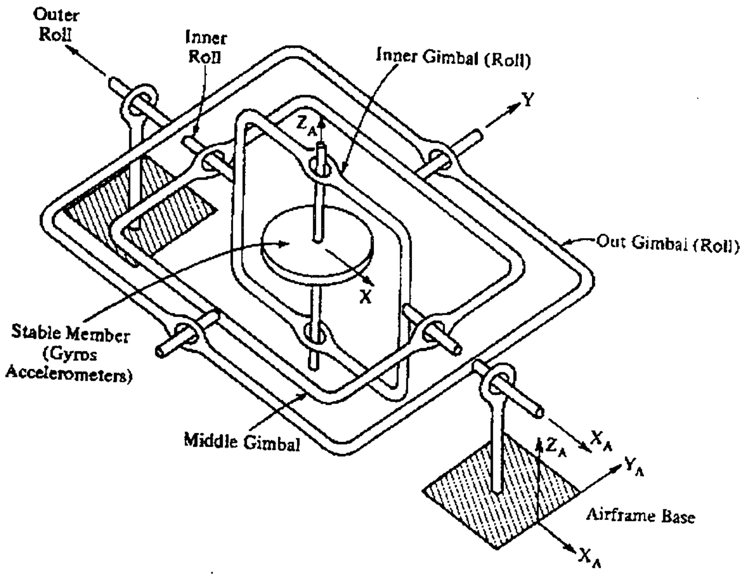


Figure 8.8: Gimbaled IMU from [Ref. 15, p. 193]

impractical for small aircraft. Also, kinematic quantities resolved in the aircraft–fixed coordinate system, necessary for stability augmentation and control, are not directly available. Instead, they must be computed through a series of Euler rotations. The computations required take time and thus introduce delays into an often time critical control problem. This fact makes the gimbaled system less than desirable for control.

2. Strapdown IMU

Strapdown IMU is conceptually the reverse of the gimbaled system mentioned above. Rather than maintaining a constant inertial orientation, a strapdown system is "strapped down" to the aircraft, thus maintaining a constant orientation in the aircraft–fixed coordinate system. Therefore, the output of the IMU is resolved in the local coordinate system. Kinematic quantities are immediately available for the control system. The extra computational burden now rests on the navigation computer which must transform the accelerations and angular velocities sensed in the local coordinate system to the inertial system. Currently, most inertial systems are of the strapdown variety. With the advent of high speed, low cost, lightweight computing power, the required calculations in transforming from the aircraft–fixed to the inertial coordinate system are no obstacle to navigation. This is the system which is modeled in this thesis.

G. INS COMPUTATIONS

In the strapdown configuration, the IMU measures angular velocities and thrust acceleration in the body frame, as well as Euler angles. However, the INS must provide position and orientation, both in the inertial frame. In order to compute position and orientation in the inertial, tangent plane coordinate system, inertial accelerations and Euler rates must be calculated. Before converting the inertial acceleration from the body frame to the inertial frame, it is necessary to compute the inertial orientation.

Computing the Euler rates is a tricky endeavor. The Euler rates are simply related to the body angular rates by Poisson's equation

$$\begin{bmatrix} \dot{\Phi} \\ \dot{\Theta} \\ \dot{\Psi} \end{bmatrix} = \begin{bmatrix} 1 & \sin\Phi \tan\Theta & \cos\Phi \tan\Theta \\ 0 & \cos\Phi & -\sin\Phi \\ 0 & \sin\Phi \sec\Theta & \cos\Phi \sec\Theta \end{bmatrix} \begin{bmatrix} p \\ q \\ r \end{bmatrix}, \tag{8.19}$$

where p, q, and r are the components of ${}^{b}\omega$, the body's angular velocity. Obviously, this formula requires the exact Euler angles. However, one can only measure these angles directly at very low frequency. At frequencies greater than a fraction of a Hertz, one must integrate the Euler rates found from Equation 8.19 to find the angles. This presents the seeming paradox of needing to know the Euler angles in order to find the Euler rates which must be integrated to find the Euler angles. This process must be implemented recursively. That is, the results of the integration to find the angles must be fed back into Equation 8.19. Furthermore, a complementary Kalman filter is necessary to provide an optimal estimate of the Euler angles, trusting the inclinometers at the low frequencies and the rate gyros at high frequencies.

Now having inertial acceleration in expressed in the body frame and the orientation of the body (Euler angles), the transformation from body to inertial can be executed. This coordinate transformation is defined by

$${}^{u}\ddot{P} = {}^{u}_{b}C\,{}^{b}a + {}^{u}g, \tag{8.20}$$

where Φ, Θ, and Ψ are the roll, pitch and yaw Euler angles, respectively, ${}^{u}\ddot{P}$ is the aircraft's acceleration in inertial coordinates, ${}^{b}a$ is thrust acceleration in the aircraft–fixed coordinate system, ${}^{u}g$ is gravity in inertial coordinates and ${}^{u}_{b}C$ is the transformation matrix from the body–fixed coordinates to inertial tangent plane coordinates as follows

$$\begin{bmatrix} \cos\Psi \cos\Theta & \cos\Psi \sin\Theta \sin\Phi - \sin\Psi \cos\Phi & \cos\Psi \sin\Theta \cos\Phi + \sin\Psi \sin\Phi \\ -\sin\Psi \cos\Theta & -\sin\Theta \sin\Phi \sin\Psi + \cos\Psi \cos\Phi & -\sin\Theta \cos\Phi \sin\Psi - \cos\Psi \sin\Phi \\ \sin\Theta & -\cos\Theta \sin\Phi & -\cos\Theta \cos\Phi \end{bmatrix}.$$

It is ${}^{u}\ddot{P}$ which can be integrated to provide velocity and position in the inertial, tangent plane frame.

H. INS ERROR SOURCES

The Inertial Measuring Unit is subject to a few main error sources. These are:

- bias
- cross–axis sensitivity
- noise floor

All of these errors apply to both accelerometers and rate gyros. Brief descriptions of these problems follow.

1. Biases

A bias in an accelerometer or a rate gyro is defined as a constant offset of the output of the device from the true value. In other words, an accelerometer might constantly read 0.01 m/s^2 while the device is not accelerating. This 0.01 m/s^2 would be referred to as a bias. This error is modeled with a series of small step functions, one for each accelerometer and rate gyro.

2. Cross–Axis Sensitivity

Cross–axis errors are caused by misalignment of the IMU with the aircraft coordinate axes. Ideally, the components of the IMU — the accelerometers and the rate gyros — would each be perfectly aligned with the three axes of the aircraft–fixed coordinate system. Unfortunately, even the highest fidelity inertial sensors are never precisely coincident with the appropriate axes. Misalignment of both types of devices causes errors. The cross–axis errors are modeled with the following equation:

$$a_{ca} = \begin{bmatrix} 0 & e_Y & e_Z \\ e_X & 0 & e_Z \\ e_X & e_Y & 0 \end{bmatrix} a, \tag{8.21}$$

where a_{ca} is the cross–axis error, e_X, e_Y, and e_Z are the cross–axis error terms and a is the actual acceleration resolved in the aircraft coordinate system. The e's from the previous equation are determined by the amount of angular offset of each sensor from the correct position.

3. Noise Floor

All sensor devices corrupt the quantities they measure. Various factors including thermal noise can cause a constant output of white noise, regardless of the actual acceleration (or angular velocity). This noise floor makes accelerations below it not measurable. That is, the output of the sensor still includes the actual quantities, but it is "invisible" because the noise floor obscures it. This process is modeled by introducing a threshold to the actual acceleration in the aircraft coordinate system and adding white noise. Taking these steps result in the accelerometer always reporting noisy, zero–mean acceleration unless the actual value is above the threshold. When the actual value exceeds the threshold, that value is added to the noise floor value (and the bias) to create the output of the accelerometer. This process also applies to the rate gyros. Now both of the sensors — GPS and INS — have been completely investigated.

IX. KALMAN FILTER DESIGN

First, recall the general process for filter design:

1. Obtain navigation equations:

$$\begin{aligned} \dot{x} &= f(x,u) \\ y &= h(x,u) \end{aligned} \tag{9.1}$$

2. Linearize 9.1 at (x_o, u_o):

$$\begin{aligned} \delta\dot{x} &= A\delta x + B\delta u \\ \delta y &= C\delta x + D\delta u \end{aligned} \tag{9.2}$$

3. Now, we need to use 9.2 to design a filter:

   i) Use 9.2 to create a synthesis model:

$$\begin{aligned} \delta\dot{x}_s &= A_s\delta x_s + B_{1s}\delta w + B_{2s}\delta u \\ \delta y_s &= C_s\delta x_s + \delta v \end{aligned} \tag{9.3}$$

where

$$\delta x_s = \begin{bmatrix} \delta x \\ \delta x_t \end{bmatrix}$$

δx_t are the states resulting from appended integrators.

$$\begin{aligned} \begin{bmatrix} \delta\dot{x} \\ \delta\dot{x}_t \end{bmatrix} &= \underbrace{\begin{bmatrix} A & K_I \\ 0 & 0 \end{bmatrix}}_{A_s} \begin{bmatrix} \delta x \\ \delta x_t \end{bmatrix} + \underbrace{\begin{bmatrix} K_p \\ I \end{bmatrix}}_{B_{1s}} \delta w + \underbrace{\begin{bmatrix} B \\ 0 \end{bmatrix}}_{B_{2s}} \delta u \\ \delta y &= \underbrace{\begin{bmatrix} C & 0 \end{bmatrix}}_{C_s} \begin{bmatrix} \delta x \\ \delta x_t \end{bmatrix} + \delta v \end{aligned} \tag{9.4}$$

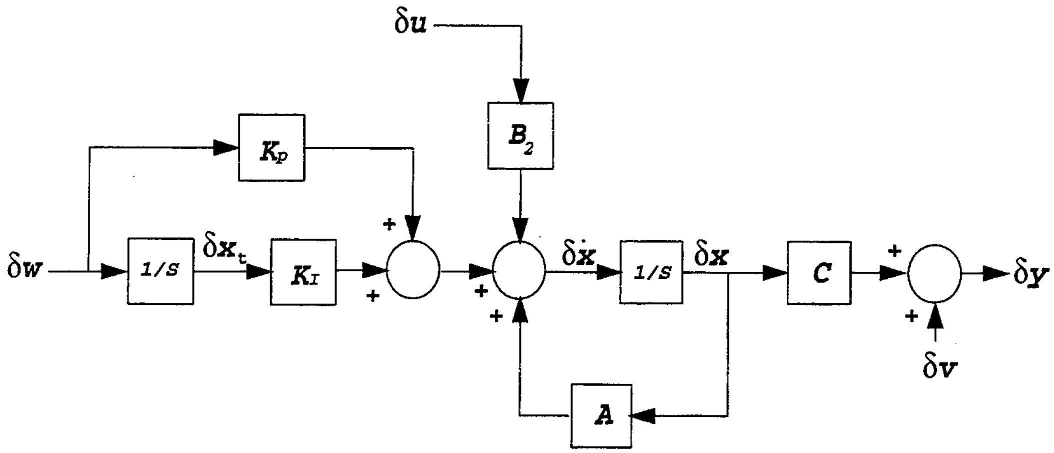


Figure 9.1: Synthesis Model

ii) Let $W = I$ and $V = I$ (intensity) where:

$$\begin{aligned} E\left(\delta w(t)\delta w(t)^T\right) &= W\delta(t-\tau) \\ E\left(\delta v(t)\delta v(t)^T\right) &= V\delta(t-\tau) \end{aligned}$$

and solve the Ricatti equation for the Kalman gain:

$$A_s X + X A_s^T - X C_s^T V^{-1} C_s X + B_{1s} W B_{1s}^T = 0$$

$$H_s = X C_s^T V^{-1} = [H \; H_I]$$

iii) Form the filter:

$$\left\{ \begin{aligned} \delta\hat{x} &= A_s \delta x_s + H_s\left(\delta y - \delta\hat{y}\right) \\ \delta\hat{y} &= C\delta\hat{x}_s \end{aligned} \right. \qquad (9.5)$$

iv) Now compute the transfer function from δy to $\delta\hat{y}$.

v) If the bandwidth is not sufficient to meet the design criteria, go back to step (ii) where a new value of V is chosen.

4. Now implement the filter on the non–linear plant:

$$\left\{ \begin{aligned} \dot{\hat{x}}_s &= f\left(\hat{x}_s, u\right) + H_s\left(y - \hat{y}\right) \\ \hat{y} &= h\left(\hat{x}_s, u\right) \end{aligned} \right. \qquad (9.6)$$

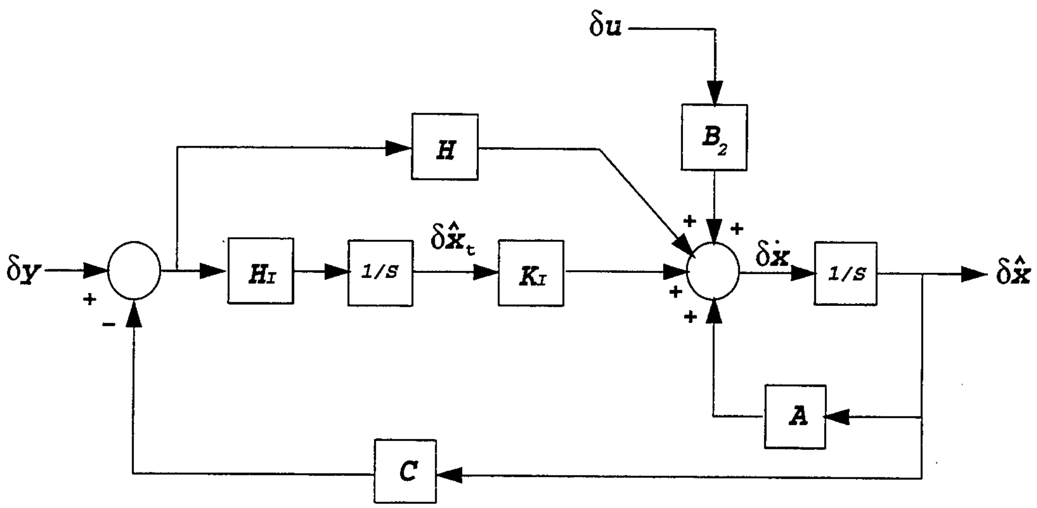


Figure 9.2: Kalman Filter Model

5. Now linearize 9.6 at (x_o, u_o) and compare the result with 9.5. They must be the same.

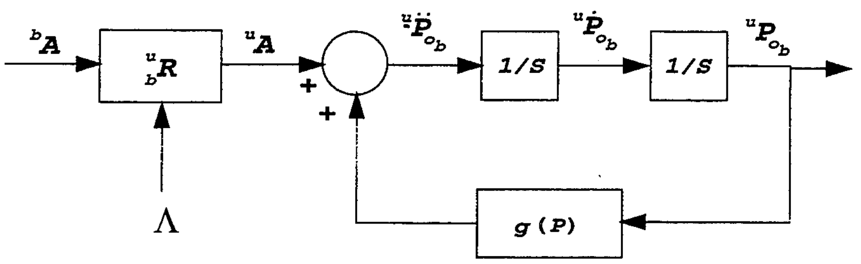


Figure 9.3: Tangent Plane Mechanization

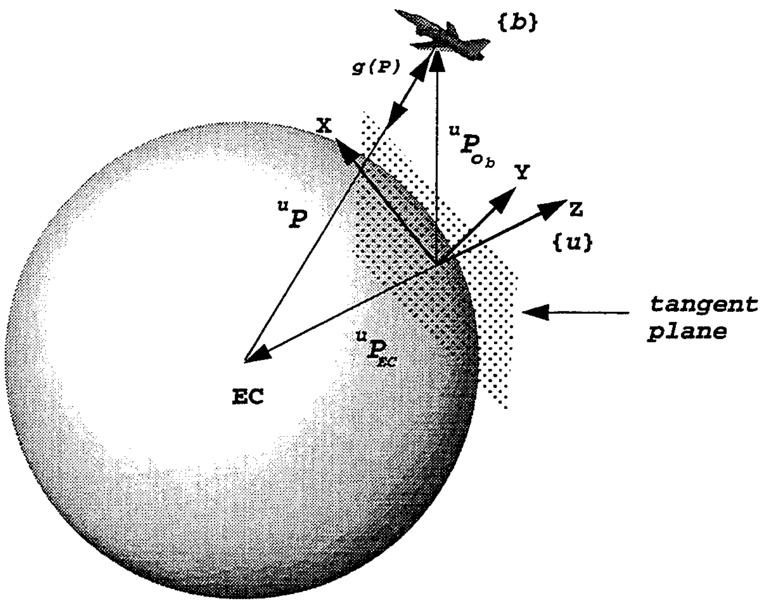


Figure 9.4: Body Frame and Tangent Plane Relationship

Now let's apply this process to Kalman filter design for tangent plane navigation.

1. Obtain navigation equations: Let

$$ {}^uP_{o_b} = \begin{bmatrix} x \\ y \\ z \end{bmatrix} \Rightarrow {}^uP_{EC} = \begin{bmatrix} 0 \\ 0 \\ -R_o \end{bmatrix} $$

$$ \Rightarrow {}^uP_{o_b} = {}^uP_{EC} + {}^uP \Rightarrow {}^uP = {}^u P_{o_b} - {}^uP_{EC} $$

$$ {}^uP = \begin{bmatrix} x \\ y \\ z \end{bmatrix} - \begin{bmatrix} 0 \\ 0 \\ -R_o \end{bmatrix} = \begin{bmatrix} x \\ y \\ z + R_o \end{bmatrix} $$

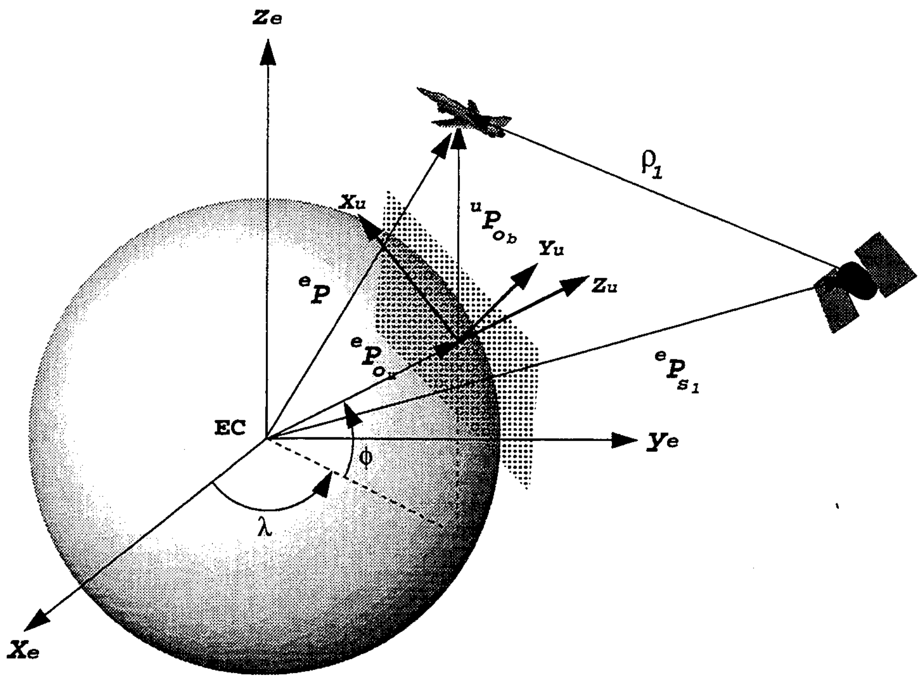


Figure 9.5: Pseudorange

Now

$$g(P) = \frac{\mu}{\|P\|^2} i_P = -\frac{\mu}{\|P\|^3} P$$

Therefore, using our notation we get

$$\left\{ \begin{array}{lcl} {}^u\dot{P}_{o_b} & = & v_{o_b} \\ \dot{v}_{o_b} & = & {}^uA + g(P) = {}^uA - \frac{\mu}{\|P\|^3} P \end{array} \right. \tag{9.7}$$

where $\dot{v}_{o_b} = {}^u P_{o_b}$. Note 9.7 is in the form

$$\dot{x} = f(x, u), \text{ where } x = \begin{bmatrix} {}^u P_{o_b} \\ v_{o_b} \end{bmatrix}$$

$$\text{and } u = {}^u A$$

Now, we need to include the pseudorange equations

$$\rho_1 = \| {}^e P - {}^e P_{s1} \| + c\Delta t \tag{9.8}$$

where $c\Delta t \equiv UERE$ (User Equivalent Range Error).

Note, all computations here must be done in $\{e\}$, since satellite positions are given in ECEF.

Let ${}^{e}_{u}R$ be given: Computing ${}^{e}_{u}R$ can be done by selecting the latitude and longitude for the origin of $\{u\}$. Then:

$$^{u}P = {}^{e}_{u}R \, {}^{u}P_{o_b} + {}^{e}P_{o_u}$$

where

$$^{e}P_{o_u} = {}^{e}_{u}R \begin{bmatrix} 0 \\ 0 \\ R_o \end{bmatrix}$$

Now,

$$\rho_x = \| \, {}^{e}_{u}R \, {}^{u}P_{o_b} + {}^{e}P_{o_u} - {}^{e}P_{sx} \| + c\Delta t \tag{9.9}$$

Finally, we need to compute Δt. Since the only real uncertainty in $UERE$ is receiver clock error, we get:

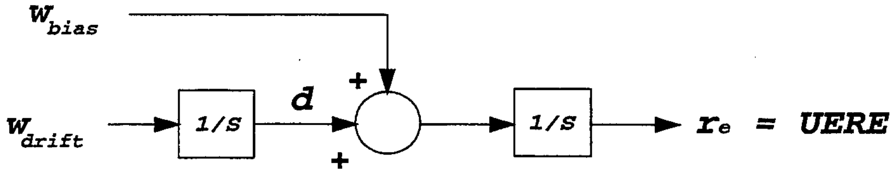


Figure 9.6: User Equivalent Range Error Mechanization

or

$$\begin{aligned} \dot{r}_e &= \Delta d + w_{bias} \\ \Delta \dot{d} &= w_{drift} \end{aligned} \tag{9.10}$$

Now collect 9.7, 9.8, 9.9, and 9.10:

$$\left\{ \begin{array}{lcl} {}^u\dot{P}_{o_b} & = & v_{o_b} \\ \dot{v}_{o_b} & = & {}^uA + g\,({}^uP) \\ \dot{r}_e & = & \Delta d + w_{bias} \\ \Delta\dot{d} & = & w_{drift} \\ \rho_x & = & \|\, {}^e_uR\, {}^uP_{o_b} + \underbrace{{}^eP_{o_u} - \,{}^eP_{sx}}_{const} \| + r_e,\ x = 1\ldots 6 \end{array} \right. \tag{9.11}$$

Notice how 9.11 is in the form:

$$\left\{ \begin{array}{lcl} \dot{x} & = & f(x,u) + \begin{bmatrix} 0 \\ I \end{bmatrix} w \\ y & = & h(x) \end{array} \right. \tag{9.12}$$

where

$$\begin{array}{lcl} x & = & \begin{bmatrix} v_{o_b} \\ {}^uP_{o_b} \\ r_e \\ \Delta d \end{bmatrix} \in R^8 \\ u & = & {}^uA \\ w & = & \begin{bmatrix} w_{bias} \\ w_{drift} \end{bmatrix} \\ y & = & \begin{bmatrix} \rho_1 \\ \vdots \\ \rho_6 \end{bmatrix} \equiv 6 \text{ pseudoranges} \end{array}$$

2. Linearize 9.11 at (x_o, u_o):

$$\begin{array}{lcl} {}^uP_{o_b} & = & P_o + \delta P,\ v_{o_b} = v_o + \delta v \\ {}^uA & = & A_o + \delta A,\ \rho_x = \rho_{xo} + \delta\rho_x \\ r_e & = & r_{eo} + \delta r,\ d = d_o + \delta d \end{array}$$

$$\left\{ \begin{array}{lcl} \delta\dot{P} & = & \delta v \\ \delta\dot{v} & = & \delta A + \left.\frac{\partial g({}^uP)}{\partial\, {}^uP_{o_b}}\right|_o \delta P \\ \delta\dot{r}_e & = & \delta d + \delta w_{bias} \\ \delta\dot{d} & = & \delta w_{drift} \\ \delta\rho_x & = & \left.\left(\frac{\partial}{\partial\, {}^uP_{o_b}} \|\, {}^e_uR\, {}^uP_{o_b} + \,{}^eP_{o_u} - \,{}^eP_{sx}\|\right)^T\right|_o \delta P + \delta r_e \end{array} \right. \tag{9.13}$$

i)

$$g({}^{u}P) = -\frac{\mu}{\left(\sqrt{x^2+y^2+(z+R_o)^2}\right)^3}\begin{bmatrix} x \\ y \\ z+R_o \end{bmatrix}$$

where

$$\begin{aligned}
{}^{u}P_{o_b} &= \begin{bmatrix} x \\ y \\ z \end{bmatrix} \\
{}^{u}P_{EC} &= \begin{bmatrix} 0 \\ 0 \\ R_o \end{bmatrix} \\
{}^{u}P &= {}^{u}P_{o_b} - {}^{u}P_{EC}
\end{aligned}$$

$$\begin{aligned}
\left.\frac{\partial g\,({}^{u}P)}{\partial\, {}^{u}P_{o_b}}\right|_o &= \left.\begin{bmatrix} \frac{\partial g_x}{\partial x} & \cdots & \frac{\partial g_x}{\partial z} \\ \vdots & \ddots & \vdots \\ \frac{\partial g_z}{\partial x} & \cdots & \frac{\partial g_z}{\partial z} \end{bmatrix}\right|_o \\
&\simeq \begin{bmatrix} -\frac{g_o}{R_o} & 0 & 0 \\ 0 & -\frac{g_o}{R_o} & 0 \\ 0 & 0 & 2\frac{g_o}{R_o} \end{bmatrix}
\end{aligned}$$

ii) It can be shown that given $y = \|RP\|$, where $R^{-1} = R^T$

$$\frac{\partial}{\partial P}y = \frac{1}{\|RP\|}R^T RP = \frac{P}{\|RP\|} \tag{9.14}$$

Using 9.14 we get

$$\begin{aligned}
\left.\left(\frac{\partial \rho_x}{\partial P_{o_b}}\right)\right|_0 &= \left.\frac{\left({}^{e}_{u}R^T\,({}^{e}_{u}R\,{}^{u}P_{o_b} + {}^{e}P_{o_u} - {}^{e}P_{sx})\right)^T}{\|\,{}^{e}_{u}R\,{}^{u}P_{o_b} + {}^{e}P_{o_u} - {}^{e}P_{sx}\|}\right|_0 \\
&= \frac{(P_o + {}^{u}_{e}R(\,{}^{e}P_{o_u} - {}^{e}P_{sx}))^T}{\|\,{}^{e}_{u}R\,{}^{u}P_o + {}^{e}P_{o_u} - {}^{e}P_{sx}\|}
\end{aligned}$$

Collecting terms, we obtain the linear error equations:

$$\begin{bmatrix} \delta\dot{P} \\ \delta\dot{v} \\ \delta\dot{r}_e \\ \delta\dot{d} \end{bmatrix} = \left[\begin{array}{c|c|cc} 0_{3,3} & I_{3,3} & 0 & 0 \\ \hline \begin{bmatrix} -\frac{g_o}{R_o} & 0 & 0 \\ 0 & -\frac{g_o}{R_o} & 0 \\ 0 & 0 & \frac{2g_o}{R_o} \end{bmatrix} & 0_{3\times3} & 0 & 0 \\ \hline 0_{1,3} & 0_{1,3} & 0 & 1 \\ 0_{1,3} & 0_{1,3} & 0 & 0 \end{array}\right] \begin{bmatrix} \delta P \\ \delta v \\ \delta r_e \\ \delta d \end{bmatrix} + \begin{bmatrix} 0 \\ I \\ 0 \\ 0 \end{bmatrix}\delta A + \begin{bmatrix} 0 \\ 0 \\ 0 \\ I \end{bmatrix}\delta w \tag{9.15}$$

$$\begin{bmatrix} \delta\rho_1 \\ \vdots \\ \delta\rho_4 \end{bmatrix} = \begin{bmatrix} \left.\frac{\partial\rho_1}{\partial\, {}^{u}P_{o_b}}\right|_0^T & 0 & 1 & 0 \\ \vdots & \vdots & \vdots & \vdots \\ \left.\frac{\partial\rho_4}{\partial\, {}^{u}P_{o_b}}\right|_0^T & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} \delta P \\ \delta v \\ \delta r_e \\ \delta d \end{bmatrix} \tag{9.16}$$

Note, Equations 9.15 and 9.16 are in the form:

$$\begin{cases} \delta\dot{x} = A\delta x + B\delta u + \begin{bmatrix} 0 \\ I \end{bmatrix} \delta w \\ \delta y = C\delta x \end{cases} \tag{9.17}$$

3. Use 9.17 to design a Kalman filter.

X. GPS/IMU INTEGRATION

Now that the kinematic equations and coordinate systems have been discussed in detail, the next step is to design the navigation system using MATLAB/*Simulink* software development tools. The aircraft chosen to evaluate the design, was the Airplane D, a medium size, high performance business like jet from Roskam [Ref. 4, Appendix C]. Mission requirements for the design were consistent with a medium–to–high altitude reconnaissance aircraft cruising in straight and level flight, taking high resolution photographs of various military and industrial sites. Flight condition 2 from [Ref. 4, Table C4] specifies a maximum weight of (13,000 lbs.), high altitude (40,000 ft) cruise, at 677 feet per second (M=0.7). At this condition, the aircraft has a pitch angle of 2.7 degrees (0.047 radians), a trim lift coefficient of 0.41 and a trim drag coefficient of 0.0335. Additional aircraft data includes a wing area of 230 square feet, a wing span of 234 feet, a mean geometric chord of 7 feet and a center of gravity (c.g.) at 0.32 mean aerodynamic chord (MAC). The stability and control derivatives used are presented in Roskam [Ref. 4, p. 620]

A. EQUATIONS OF MOTION

The first step in the design of the integrated system is to develop the appropriate equations of motion (EOM) for the aircraft. A right–handed body coordinate system $\{b\}$ was chosen referenced to a universal frame $\{u\}$ with a NED orientation. The derivation of the EOM's starts with the following four equations:

$$\begin{aligned}
\frac{d}{dt}V_b &= -{}^b\omega_b \times {}^bV_b + \frac{{}^bF}{m} \\
{}^b\dot{\omega}_b &= -I_b^{-1}\left({}^b\omega_b \times \left(I_b\,{}^b\omega_b\right)\right) + I_b^{-1}\,{}^bN \\
\dot{\Lambda} &= S(\Lambda)\,{}^b\omega_b
\end{aligned}$$

$$^{u}\dot{P}_{b} = {}^{u}_{b}C\,{}^{b}V_{b}$$

where

$$\begin{aligned} F &= \text{Forces} \\ N &= \text{Moments} \end{aligned}$$

The force and moment terms consist of the thrust, aerodynamic and gravity terms. Assuming the moment about the c.g. due to gravity is zero, the force and moment equations reduce to:

$$\begin{aligned} F &= F_T + F_A + F_g \\ N &= N_T + N_A \end{aligned}$$

where

$$\begin{aligned} T &= \text{Thrust} \\ A &= \text{Aerodynamic} \\ g &= \text{Gravity} \end{aligned}$$

The thrust terms are computed from the engine thrust settings and aircraft geometry and the gravitational force is computed by resolving the gravitational constant $32.174\frac{ft}{sec^2}$ from the universal to the body frame.

The aerodynamic forces are calculated using a first order approximation of the stability and control derivatives contained in Roskam [Ref. 4]. Starting with fundamental aerodynamic force equation:

$$F = \frac{1}{2}\rho V^2 S C_F \qquad (10.1)$$

where

$$F = [-D, -Y, -L, l, m, n] \qquad (10.2)$$

the first order linear approximation becomes:

$$F = \bar{q}\bar{S}\left(C_{f1} + \frac{d}{dx'}\left(C_F\right)\left(x' - x'_o\right) + \frac{d}{d\dot{x}'}\left(C_F\right)\left(\dot{x}' - \dot{x}'_o\right) + \frac{d}{d\Delta}\left(C_F\right)\left(\Delta - \Delta_o\right)\right)$$

where

$$\begin{aligned}
x &= \begin{bmatrix} u & v & w & p & q & r \end{bmatrix}^T \\
x' &= \frac{1}{V_t}\begin{bmatrix} u & v & w & \frac{bp}{2} & \frac{\bar{c}q}{2} & br \end{bmatrix}^T \\
\dot{x}' &= \begin{bmatrix} \dot{u}' & \dot{\beta}' & \dot{\alpha}' & \dot{p}' & \dot{q}' & \dot{r}' \end{bmatrix}^T \\
\bar{S} &= diag\begin{bmatrix} -S & S & -S & Sb & S\bar{c} & Sb \end{bmatrix} \\
\Delta &= \begin{bmatrix} \delta e & \delta a & \delta r & \delta T \end{bmatrix}^T
\end{aligned}$$

The initial development of the EOM involves only the $\frac{d}{dt}{}^bV_b$ and $\frac{d}{dt}{}^b\omega_b$ equations. In matrix form this becomes the basic EOM equation:

$$\begin{aligned}
\begin{bmatrix} \frac{d}{dt}{}^bV_b \\ \frac{d}{dt}{}^b\omega_b \end{bmatrix} &= \begin{bmatrix} -{}^b\omega_b \times {}^bV_b \\ -I_b^{-1}\left({}^b\omega_b\left({}^b\omega_b \times \left(I_b \times \left(I_b\,{}^b\omega_b\right)\right)\right)\right) \end{bmatrix} + \begin{bmatrix} {}^b_uC\,{}^ug \\ 0 \end{bmatrix} \\
&+ \begin{bmatrix} T_o\frac{\delta T}{m} \\ 0 \end{bmatrix} + \begin{bmatrix} {}^b_wC & 0 \\ 0 & {}^b_wC \end{bmatrix}\begin{bmatrix} {}^wF_A \\ {}^wN_A \end{bmatrix}
\end{aligned}$$

where I_b is the aircraft's inertia tensor and the aerodynamic forces and moments are computed in the wind reference frame $\{w\}$ and must be converted to the body frame $\{b\}$. The first order approximation of the aerodynamic forces and moments was substituted into the basic EOM equation and the Λ equation was added yielding the complete EOM equation. After the trim values were computed, they were included as initial conditions in the EOM's integrator. The diagram was then linearized, using the MATLAB linmod.m command. EOM verification was done by comparing the eigenvalues of the linearized model to Roskam's data. The results are shown in Table 10.1.

TABLE 10.1: EIGENVALUES

	Roskam	EOM
Long period	$-0.00593 \pm 0.0904i$	$-0.0103 \pm 0.0912i$
Short period	$-0.9943 \pm 2.641i$	$-0.7941 \pm 2.714i$
Spiral	-0.0012	$+0.0007$
Dutch roll	$-0.05848 \pm 1.684i$	$-0.0567 \pm 1.686i$
Roll mode	-0.5023	-0.5036

The linearized values compare quite nicely to Roskam's values except for one unstable eigenvalue in the lateral dynamics. This pole corresponds to the spiral mode and has a large time constant. It does not affect stability in the longitudinal dynamics. Finally, the three position states were added to the initial EOM model to make the final EOM model, shown in Figure 10.1, which is used as an input to the IMU System.

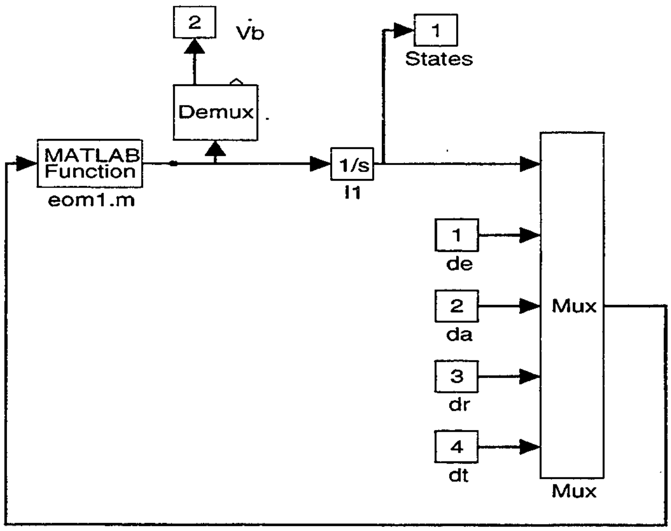


Figure 10.1: Equations of Motion *Simulink* Block Diagram

B. INERTIAL MEASUREMENT UNIT (IMU)

In order for the IMU to calculate position and velocity in the universal frame u, the aircraft's angular rates about the x, y, and z axes (p, q, and r respectively) and thrust accelerations in a_x, a_y, and a_z, must be measured. A real IMU uses angular gyros and accelerometers to provide this data, where the design IMU model, Figure 10.2, uses state vector outputs from the EOM model to perform the acceleration calculations. The IMU inputs consist of the linear velocity components (u, v, w) or V_b , angular rates (p, q, r) or ω_b , Euler angles (ϕ, θ, ψ) or Λ , and the position of the body in the universal frame, ${}^uP_b$. Additionally, the linear accelerations, $\dot{V}_b$, are also provided to complete the calculations. Given the small size of the aircraft, it was assumed that the IMU was located at the aircraft's center of gravity and all calculations are based on this assumption. Since real IMU sensors possess some inherent

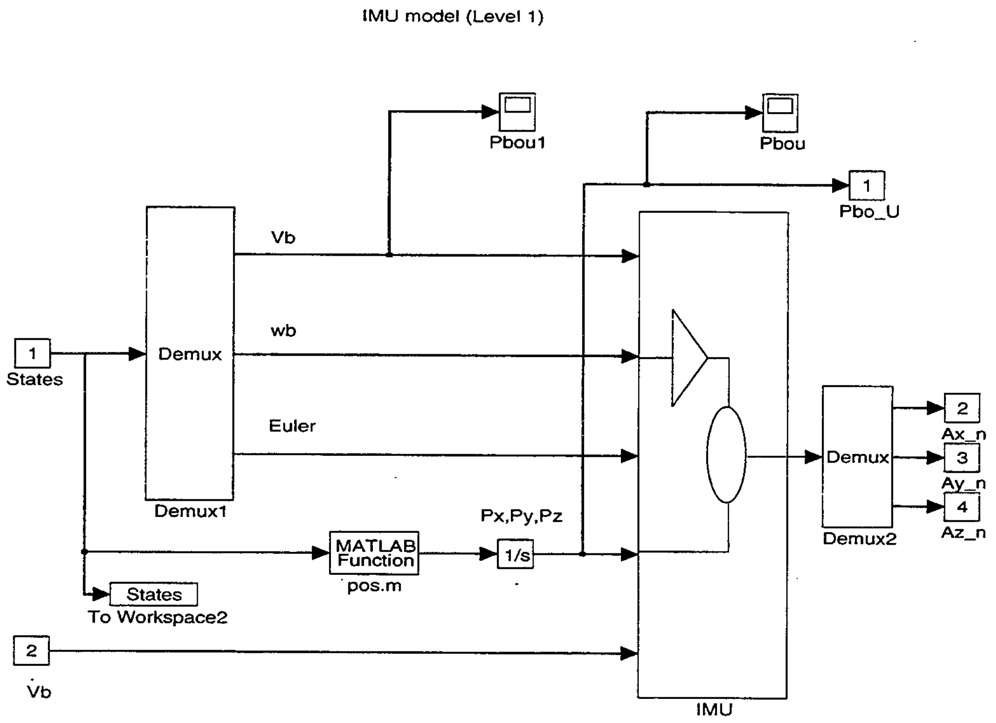


Figure 10.2: Inertial Measurement Unit (Level 1) *Simulink* Block Diagram

noise and bias characteristics, white gaussian noise was included to simulate the noisy sensors and a nominal bias to account for the design limitations of the sensors.

To calculate the accelerations in the $\{u\}$ frame, Figure 10.3, the gravity force

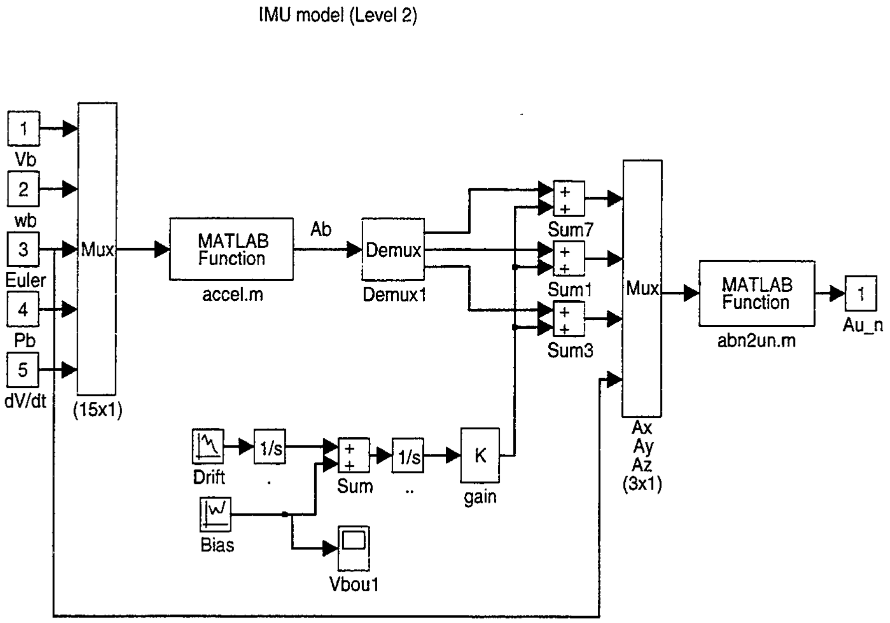


Figure 10.3: Inertial Measurement Unit (Level 2) *Simulink* Block Diagram

was calculated and a transformation matrix from $\{u\}$ to the body frame $\{b\}$ was applied:

$$^{b}g = {}^{b}_{u}C\,{}^{u}g \tag{10.3}$$

The accelerations in the body frame where calculated and the acceleration due to gravity was removed:

$$^{b}a = \frac{d}{dt}\,{}^{b}V_{b} + {}^{b}\omega_{b} \times {}^{b}V_{b} - {}^{b}g \tag{10.4}$$

The MATLAB routine to perform these calculations was accel.m and is provided in Appendix B. After the accelerations in the body frame were calculated, the noise and bias errors were incorporated to simulate actual acceleration measurements from the IMU. Since they were calculated in the body frame, the noisy accelerations, ${}^{b}a_n$, were transformed to the universal frame using a transformation matrix, ${}^{u}_{b}C$. The MATLAB function used to perform this calculation was abn2un.m. The resulting outputs from the IMU were noisy accelerations in the universal frame, ${}^{u}a_n$.

C. GLOBAL POSITIONING SYSTEM (GPS)

In order to provide accurate position information that could be used to update or verify the position estimates calculated from the IMU accelerations, a GPS mechanization was designed. The GPS mechanization, Figure 10.4, was designed to provide a pseudo–range measurement based on the aircraft's estimated position and the position of four satellites. Inputs to the GPS came from the EOM linear velocity terms, V_b , which were transformed from the body frame to the universal frame and integrated to provide a position estimate, ${}^{u}P_b$. Satellite positions were input as a constant value using a satellite initial position MATLAB function, initsat.m. A

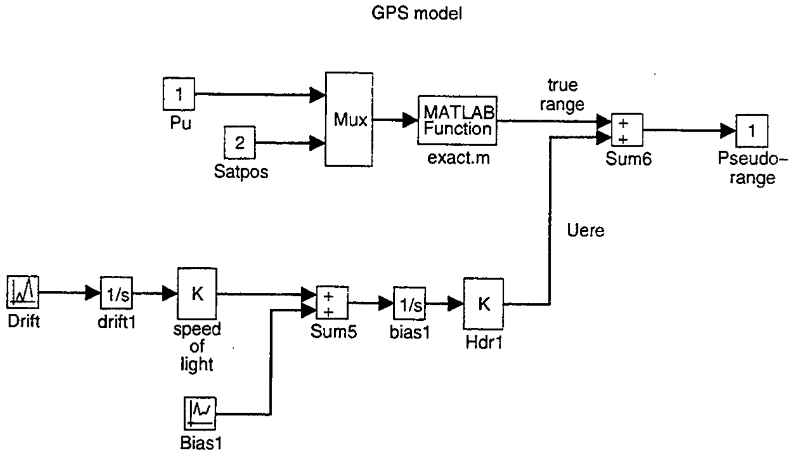


Figure 10.4: GPS Pseudo–range *Simulink* Block Diagram

MATLAB function, exact.m, calculated the exact ranges from each satellite. User equivalent range error (UERE) was determined by introducing white gaussian noise to simulate the GPS receivers inherent noise bias and clock drift. Combining the exact range with the UERE, a pseudo range value was calculated. Along with the noisy accelerations calculated in the IMU, the pseudo–range values were input to the Kalman filter.

D. KALMAN FILTER

The Kalman filter was designed for the purpose of providing optimal inertial velocities and positions given noisy accelerations from the IMU and pseudo–ranges from a GPS receiver. To accomplish this, it was desired to have a break frequency of $1\frac{rad}{sec}$ as the cutoff below which only GPS information would be valid. This constituted steady–state, straight and level flight. For a maneuvering aircraft, frequencies greater than $1\frac{rad}{sec}$, the sampling rate of the GPS would not be capable of providing adequately up–dated position information. At this point, the accelerometer data would be considered valid. A design criterion of $1\ \frac{rad}{sec}$ as a -3dB cutoff for the frequency response was chosen to meet these requirements.

1. Navigation Model

The first step in the filter design was to develop a navigation model which provided velocity and position as outputs. Using the position outputs, pseudo–ranges could be calculated. The navigation model was divided into two components. The main component, Figure 10.5, received the noisy accelerations from the IMU which were then integrated twice to produce velocity and position in the universal frame. The second component, Figure 10.6, calculated estimated pseudo–ranges to the four satellites given their position, the aircraft's estimated inertial position based on the accelerations, and a clock drift error component. The output of the navigation model

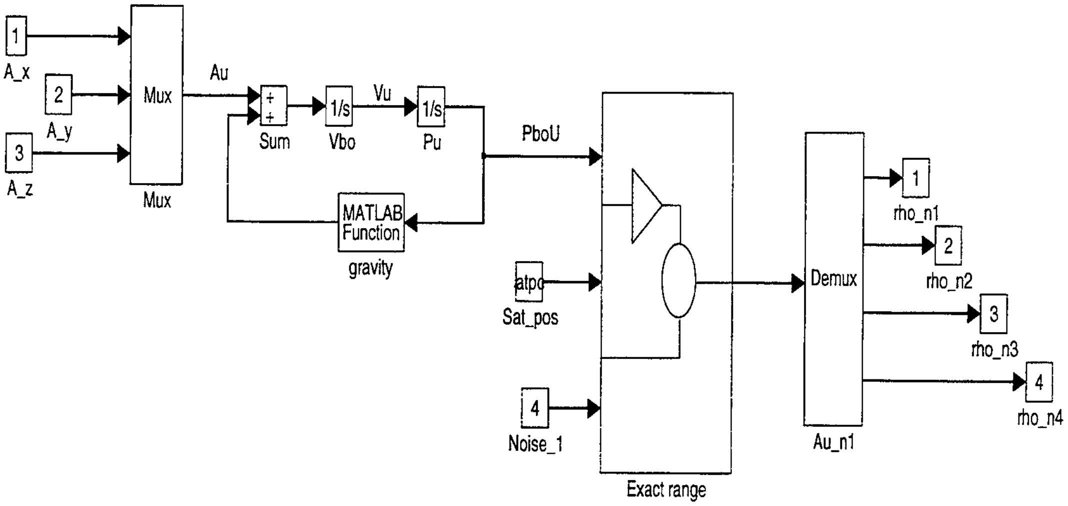


Figure 10.5: Navigation Model *Simulink* Block Diagram

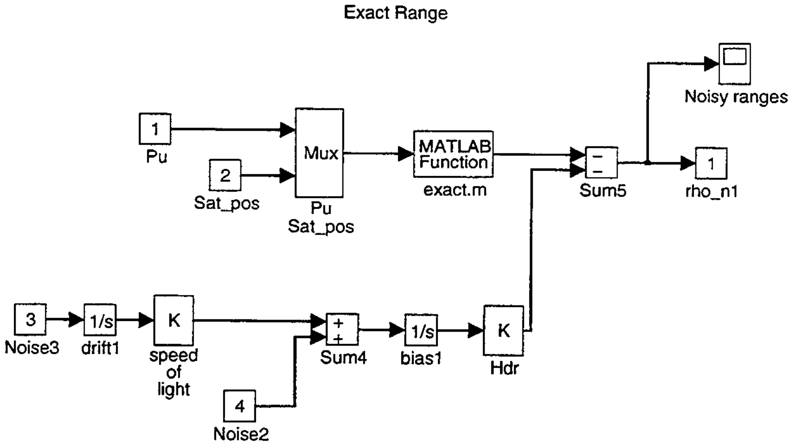


Figure 10.6: Kalman Filter, Pseudo-range *Simulink* Block Diagram

was four estimated pseudo–ranges. These estimated ranges were compared to actual pseudo–ranges from the GPS receiver.

2. Linearized Model

The next step was to linearize the navigation model at a nominal aircraft position. This was accomplished by using the MATLAB function, linmod.m, on the navigation model. The linearized state space matrices, $[A, B, C, D]$, were then used in the development of the synthesis model. In state space form, the linearized equations become:

$$\begin{aligned} \delta\dot{x} &= A\delta x + B\delta u \\ \delta y &= C\delta x + D\delta u \end{aligned}$$

Here $D = 0$.

3. Synthesis Model

The next step in the design of the Kalman filter, was to create a sythesis model which accounted for measurement and sensor noise inputs. In this case only sensor noise bias was considered. By considering only noise *bias* in each accelerometer axis, only one integrator per axis is required. These appended integrators result in three poles at the origin. As the gains are increased, the poles will diverge in three directions. One on the negative real axis and the other two will depart at 45 degree angles out the left–half plane. To insure system stability, three zeros are included so that the poles converge to a value along the negative real axis. In the synthesis model, zeroes were chosen at -1, which forces the poles to converge on the negative real axis. In addition to the noisy inputs, the accelerations in each axis are included as inputs. The resulting synthesis model is shown in Figure 10.7. Next, the synthesis model is linearized to determine the synthesis state space matrices, $[A_s, B_s, C_s, D_s]$. The resulting state space form is:

$$\begin{aligned} \delta\dot{x}_s &= A_s\delta x_s + B_{1_s}\delta w + B_{2_s}\delta u \\ \delta y &= C_s\delta x + \delta v \end{aligned} \tag{10.5}$$

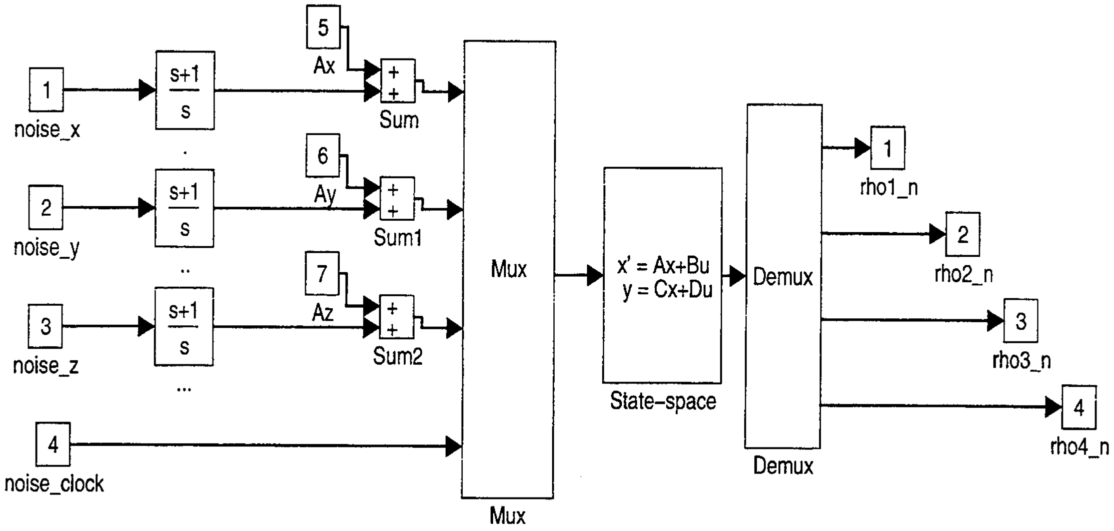


Figure 10.7: Synthesis Model for Kalman Filter, *Simulink* Block Diagram

In the equation 10.5, δv is zero since only input noise is considered.

4. Noise Intensity and the Non–Linear Plant

In order to solve the Ricatti equation 10.6 for the optimal gain

$$A_s X + X A_s^T - X C_s^T V^{-1} C_s X + B_{1_s} W B_{1_s}^T = 0 \tag{10.6}$$

the noise intensities V and W must be adjusted to provide the optimum frequency response. The initial values were chosen to be identity matrices, I. Once the values were chosen, the MATLAB function, lqe.m, was used to determine the optimal Kalman gain, H_s. The Kalman gain H_s was partitioned based on the arrangement of the states in the B_s matrix. The partitioned Kalman gains were implemented on the non-linear plant shown in Figure 10.8

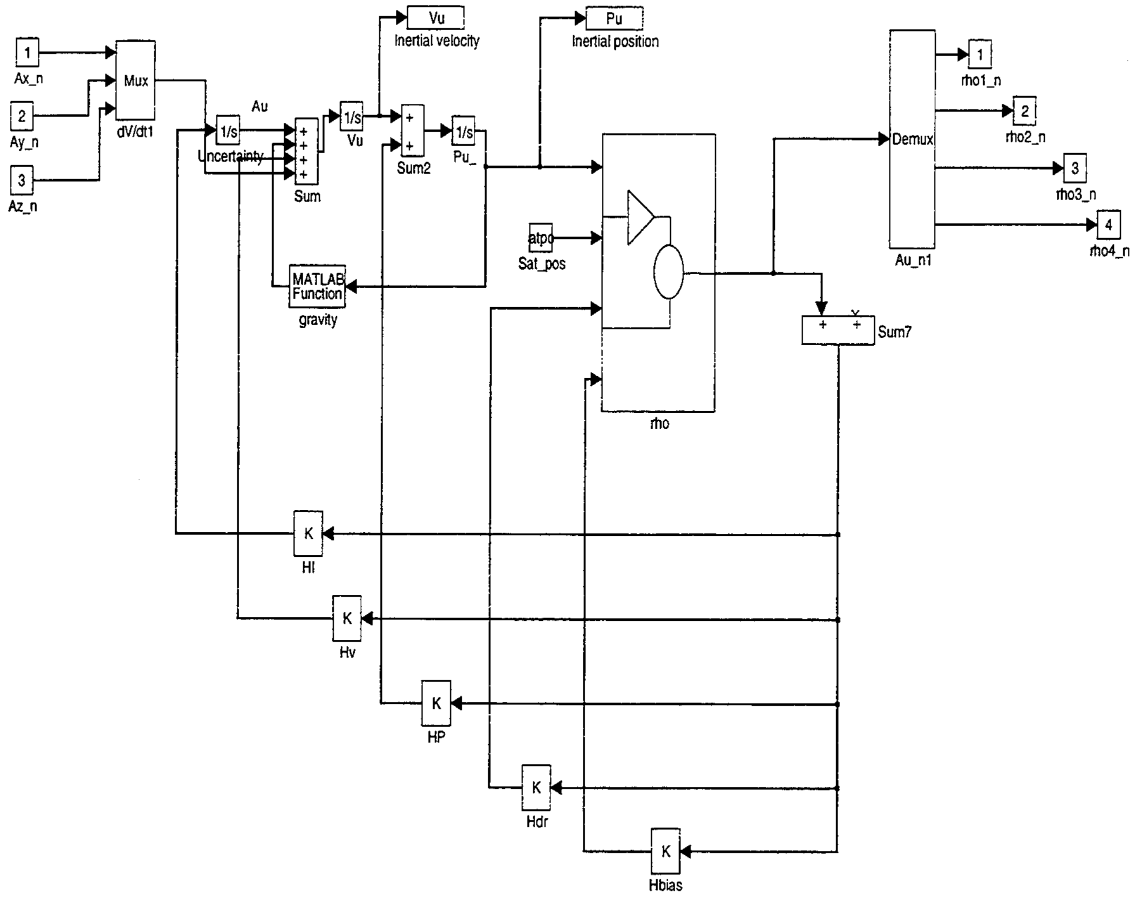


Figure 10.8: Non-Linear Kalman Filter

Next, a bode plot of the frequency response was generated and the bandwidth's were observed. The process was repeated using a function called, kal.m. This allowed the weighting factors to be adjusted until the optimum frequency response was obtained. It was necessary to decrease the weighting factor on the outputs V and increase the weighting on the inputs W. The optimal bandwidth of $1 \frac{rad}{sec}$ was acheived at V and W values of $0.05I$ and $6I$, respectively. The frequency responses from the pseudo range to the pseudo range estimates are shown in Figures 10.9 and 10.10. It can be seen that the -3dB cutoff occurs at exactly 1 $\frac{rad}{sec}$ for each input. The frequency responses of the filter to accelerations in the x, y and z directions are shown in Figure 10.11. The 20dB per decade slope of the curve below the frequency of 0.1 $\frac{rad}{sec}$, indicates the accelerometer inputs are washed out at low frequencies. This is within the design criteria. To verify the response of the filter to a disturbance in acceleration, a step input was performed in each channel and the pseudo–range responses are shown in Figure 10.12.

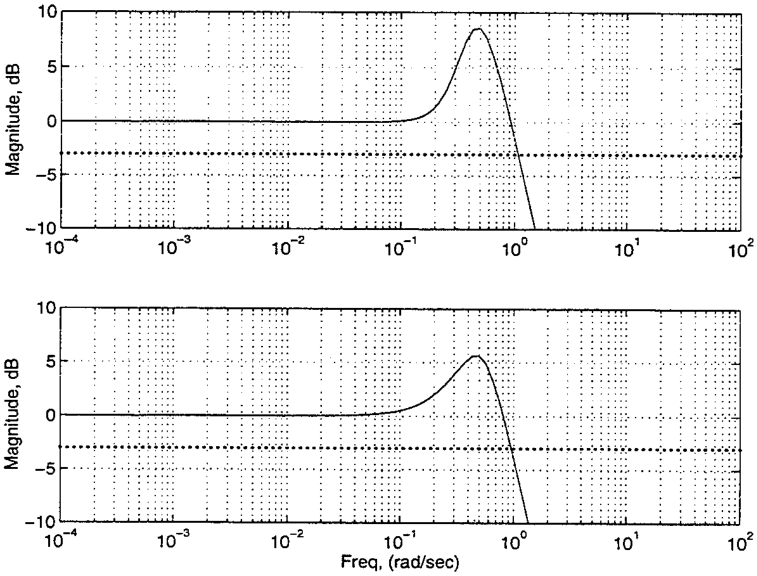


Figure 10.9: Frequency Response, Pseudo-range to Pseudo-range Estimate 1 & 2

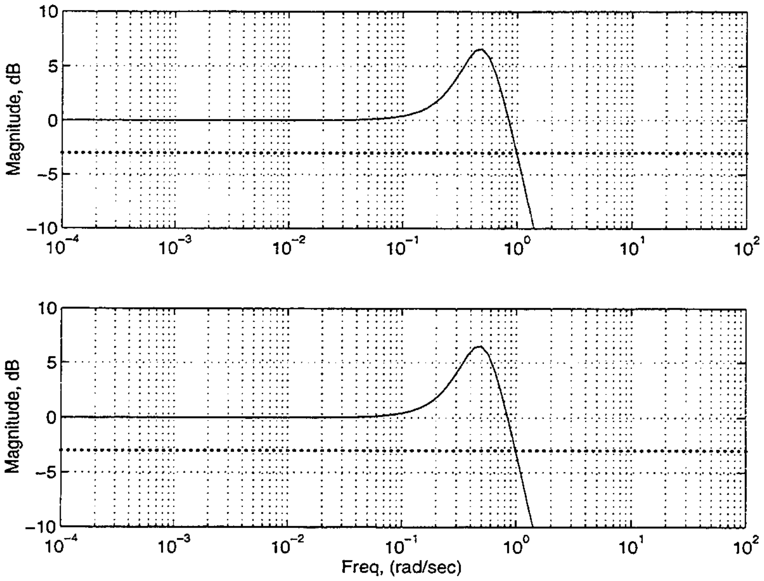


Figure 10.10: Frequency Response, Pseudo-range to Pseudo-range Estimate 3 & 4

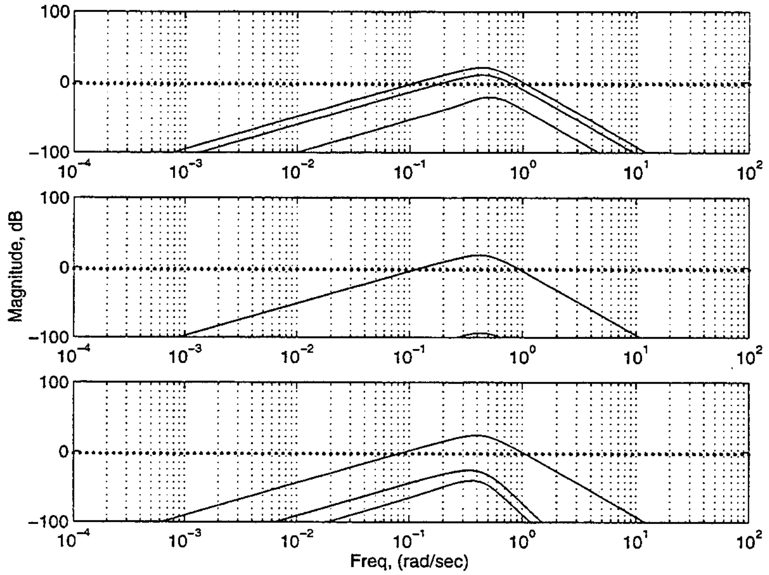


Figure 10.11: Frequency Response, Accelerations to Pseudo-range Estimate

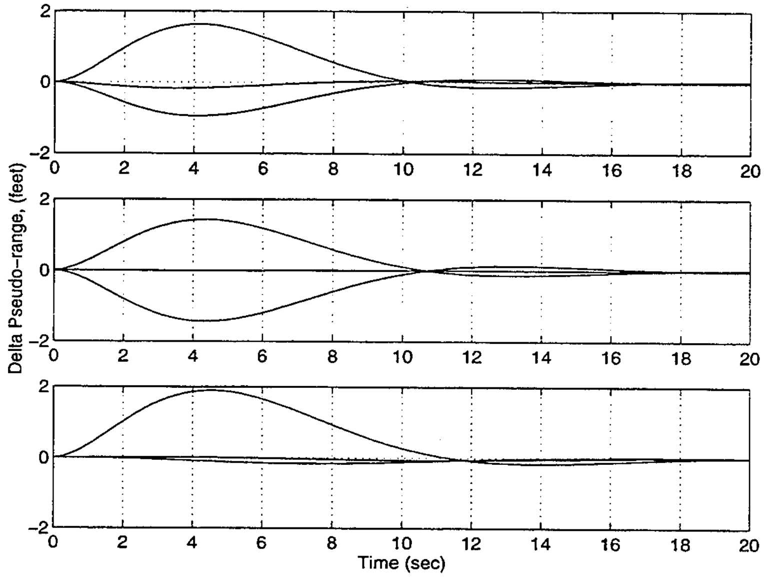


Figure 10.12: Pseudo–range Response to Step Input in A_x, A_y, A_z

XI. CONCLUSION

The purpose of this thesis has been to provide the engineer with a solid physical and mathematical understanding of the relationship between the GPS/IMU sensors and the air vehicle and how to integrate the two systems properly.

The extensive use of 3–D graphics for depicting the coordinate relationships and the corresponding transformations, gives meaning to the physical and mathematical relationships between the two systems. By introducing the fundamental concepts behind inertial sensors and global positioning systems, the data each provides can be understood and integrated properly. Next, the errors that are common to both systems are discussed and error analysis is applied to reduce the effects on the output of the integrated system. Introducing the design method for Kalman filtering allows the engineer to develop a system which accounts for the inherent errors and guarantees positional accuracy no matter which sensor is providing the information. Developing the design through the use of MATLAB/*Simulink* software, allows the engineering team the opportunity to apply all of the concepts introduced in this thesis and gives them the flexibility to expand on the basic concepts.

This thesis has provided the uniform framework required in the design of an integrated GPS / IMU system using Kalman Filtering. As GPS and IMU components become smaller and more lightweight, their use in Unmanned Autonomous Vehicle's (UAV's) will increase dramatically. The need for precise navigational data will continue to rise as the requirements for UAV's increase to meet their predicted threat. Providing the engineer with a complete document which approaches the design process in a step–by–step manner, is essential if design team is to compete effectively.

APPENDIX A: SHOWCASE, SNAPSHOT, AND XV UNIX SOFTWARE

The graphical models used in the development of this thesis were produced using the Silicon Graphics Indigo computers available in the Aeronautical Engineering Computer Laboratory and the Avionics Design Laboratory, both located in Halligan Hall. The software utilized to produce the 3–D models was IRIS Showcase (version 3.3).

A. IRIS SHOWCASE 3.3

To access the Showcase graphical program, type "showcase" at the prompt in the winterm window. Three windows will appear when "Showcase" comes on–line. The first and largest window will be the drawing tablet, figure A.1. This window is used for producing the actual picture. The next window is called the "Showcase

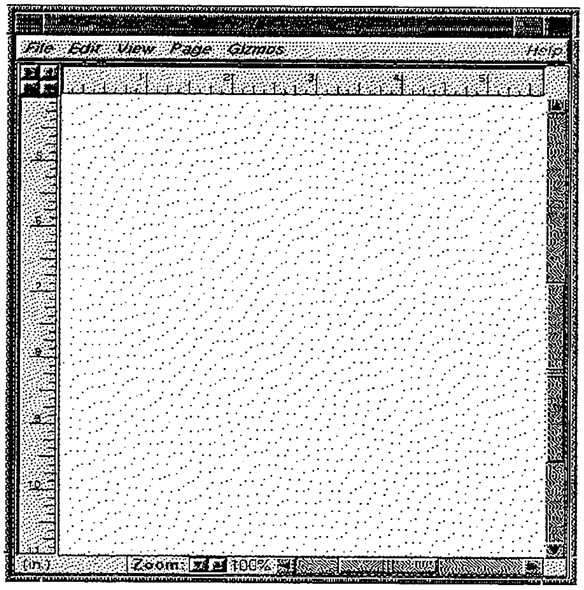
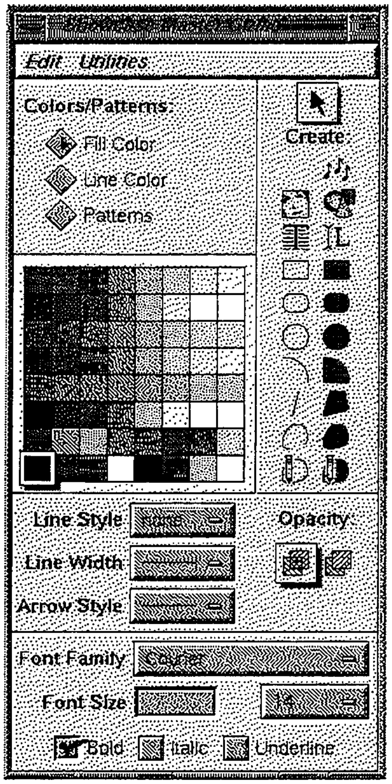


Figure A.1: Showcase Drawing Tablet

Master Gizmo". This is the main tool for selecting 1-D drawing devices, figure A.2. The third window is a "status gizmo", figure A.3, which displays the status of each

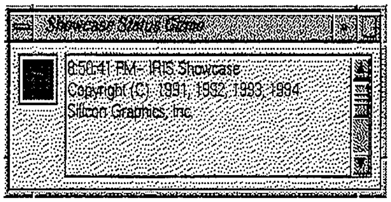


Font Family: Courier

Font Size: 14

Figure A.2: Showcase Master Gizmo

command selected or action performed in "Showcase".



8:50:41 PM - IRIS Showcase
Copyright (C) 1991, 1992, 1993, 1994
Silicon Graphics, Inc.

Figure A.3: Showcase Status Gizmo

1. Creating 1–D and 3–D Objects

To draw 1-D figures, select the appropriate tool from the "Master Gizmo" using the mouse "point and click" method. The method is similar to all graphics programs used on MacIntosh and IBM computers. Type styles can be selected from the same tool-box and fonts can be chosen at the lower portion of the "Master Gizmo" window. These operations are self–explanatory.

To draw 3-D objects, the strength of this program, go to the "Gizmo" pull down menu at the top of the drawing tablet. Select the "3–D" option from the menu and "3D Gizmo", figure A.4, will appear. Using the mouse arrow, select the "Create" tool in the upper right corner of the "3D Gizmo" window. Next, place the mouse arrow in the drawing area of the tablet. A red box will appear and move about the window as the pointer is moved. Place the window in a selected position and click the left mouse button to drop the box. A 3D room will appear on the page. This is the drawing surface for the 3D object, figure A.5. Next, return to the "3D Gizmo" and select any of the Simple models using the "point and click" method.

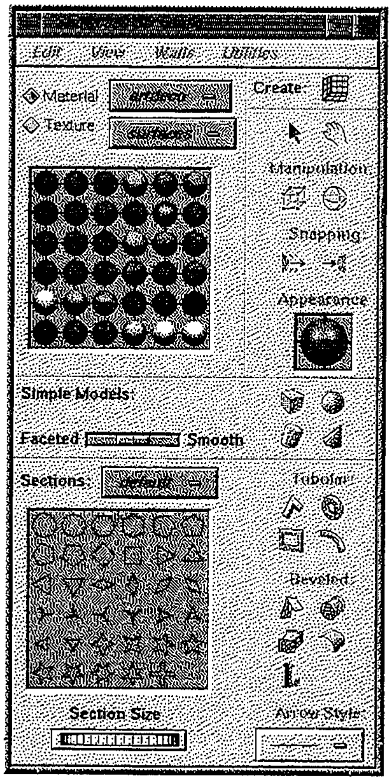


Figure A.4: Showcase 3D Gizmo

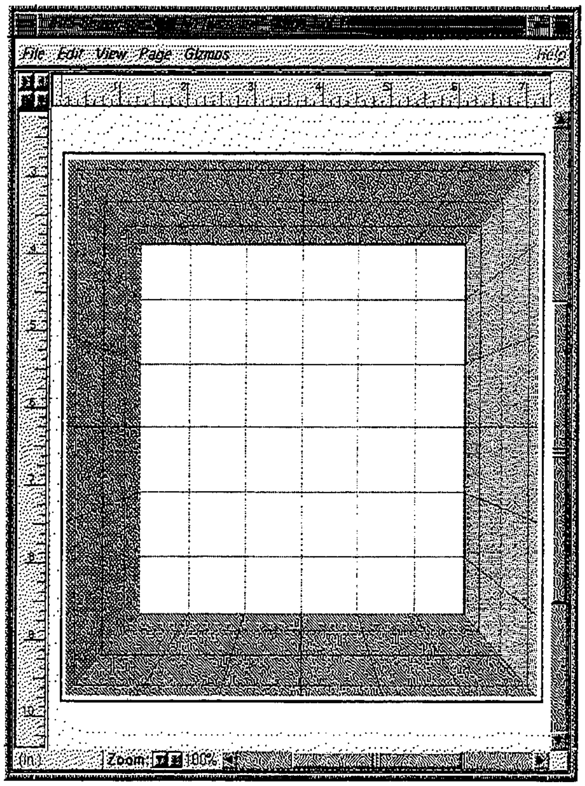


Figure A.5: Showcase 3D Drawing Window

Tubular and beveled surfaces can be chosen also. The Faceted and Smooth sliding scale tool adjusts the number of surfaces the object will have. The smoother the surface, the more memory the object will require. Moving the mouse arrow back to the 3D drawing window, place the pointer in the appropriate spot and hold the left mouse button down while dragging the object outward, figure A.6. In the case of a sphere, the size of the radius depends on how far the pointer has been dragged from the initial position. Selecting the Material and Texture of the object produced is self explanatory and well worth the time spent experimenting. Once the 3–D object has been produced, the orientation of the object within the 3–D drawing window can be manipulated using either the "hand" tool on the "3D Gizmo" window or by placing the mouse arrow on any surface "wall" inside the 3–D drawing window and holding left mouse button down while moving the mouse. The room will rotate to the desired

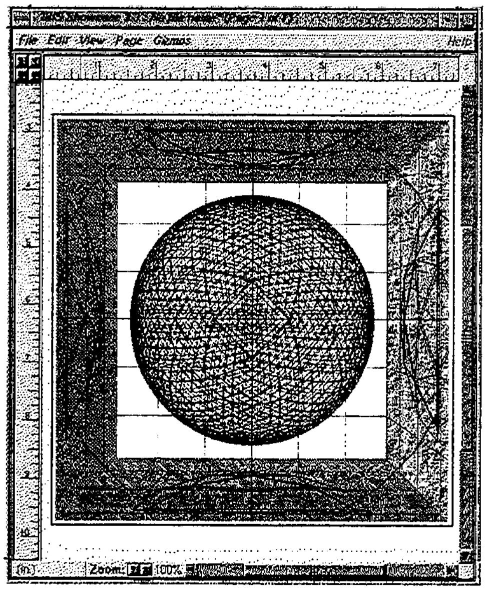


Figure A.6: Showcase 3D Sphere

orientation.

Once the 3–D model has been produced, place the mouse arrow outside of the 3–D drawing window and click the left mouse button. The drawing window will disappear and 1–D objects can once again be produced on or around the 3–D object, figure A.7.

2. Saving the Drawing

The object can be saved in a variety of formats. The default style is a "Showcase" document. All files should be saved in this format so that they can edited at a later date. Encapsulated Postscript (EPS) and Postscript (PS) styles are the style of choice, for they can be imported into a variety of word processing programs. However, some programs do not allow the operator to choose the size of the object when it is imported into the document. Since this thesis was produced using LaTeXtypesetting software, both styles will be discussed.

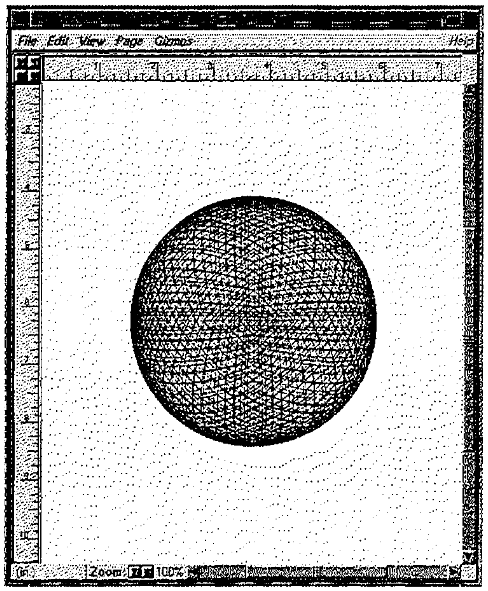


Figure A.7: Showcase 3D Sphere in 1D Environment

3. Encapsulated Postscript

Once the object has been created, return to the pull down menus on the drawing tablet. Select the "Edit" pull down menu and choose the "Select All" function. This highlights all objects in the drawing window. Next, choose the "File" pull down menu and select the "Save as" function. Moving the mouse right an additional pull down menu will allow the operator to select the type of file format which the object will be saved as. Choose the "...as EPS" function. A "Showcase File Browser" window will appear and the file name can either be chosen or typed in at the bottom of the window, figure A.8. The only problem with importing EPS files into a LaTeXdocument is that the size of the object cannot be adjusted using "height and width" commands where they can be if a PS file is used.

4. Postscript

The same procedure is used for saving the object as a PS file. However, when saving a Postscript file, the default device the file will be saved to is the printer. An additional menu will appear which will allow the operator to choose whether to

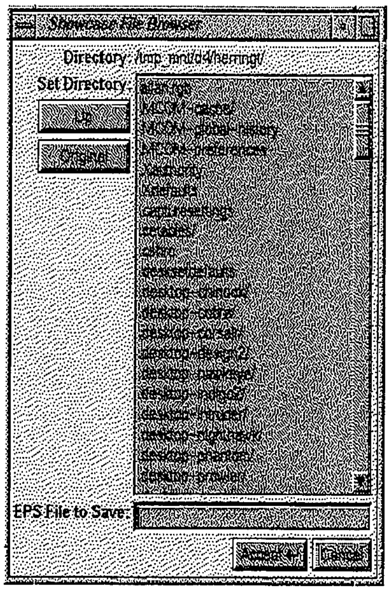
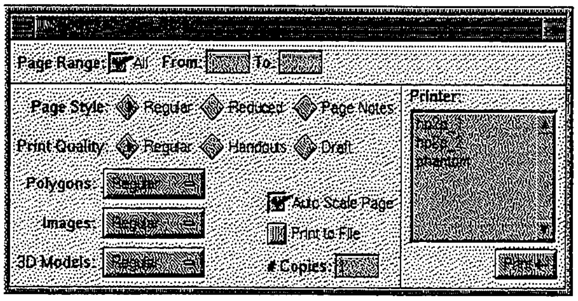


Figure A.8: Showcase File Browser

print the object to a file or the printer, figure A.9.


Page Range: ☑ All From: To:

Page Style: ◆ Regular ◇ Reduced ◇ Page Notes

Print Quality: ◆ Regular ◇ Handouts ◇ Draft

Polygons: Regular

Images: Regular

3D Models: Regular

☑ Auto Scale Page

□ Print to File

Copies: 1

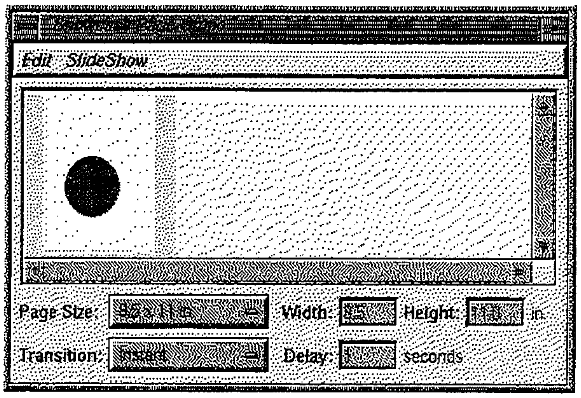
Figure A.9: Showcase PS Save Window

5. Additional 3D Goodies

A slide show presentation function is available by selecting the "Page Gizmo" from the "Page" pull down menu on the main drawing tablet, figure A.10. Experimentation is the best way to learn the system!

B. SNAPSHOT

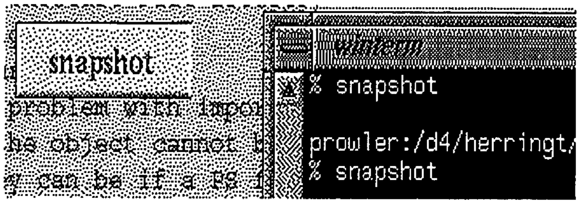
Including graphics from the display requires that they be converted into a Postscript file. To do this, the SGI computers have the capability to take a "snap-



Page Size: 8.5 x 11 in. Width: 8.5 Height: 11.0 in.
Transition: Instant Delay: 1 seconds

Figure A.10: Showcase Page Gizmo

shot" of the graphic and save it as a "*.rgb" file. This file extension is the default to the SGI system. This is accomplished by typing "snapshot" at the winterm window. A small window will appear in the upper left portion of the display, figure A.11. Place the mouse arrow on the "Snapshot" window and depress the left mouse button.



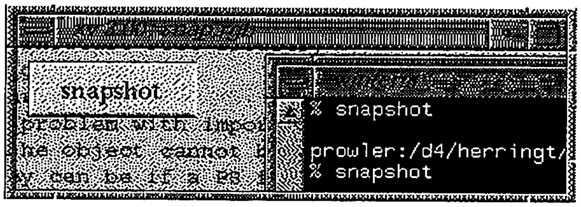
```
% snapshot

prowler:/d4/herringt/
% snapshot
```

Figure A.11: SGI Snapshot Window

Drag the mouse over the object that is to be copied. A red box will position around the object. The box size can only be controlled when the mouse arrow is on the "Snapshot" window. If the edge of the box does not move, the mouse arrow must be re–positioned on the window. Once the object has been boxed, place the mouse arrow on the "Snapshot" window and depress the right mouse button. A small window will appear which will allow the object to be saved as a "*.rgb" file. Name the file accordingly and then "save" or "save and exit" the program.

C. XV

Once the object has been saved as an "*.rgb" file, it can be converted to Postscript format by using the "XV" program on the SGI computers. Type "xv <filename.rgb>" at the in figure A.12.

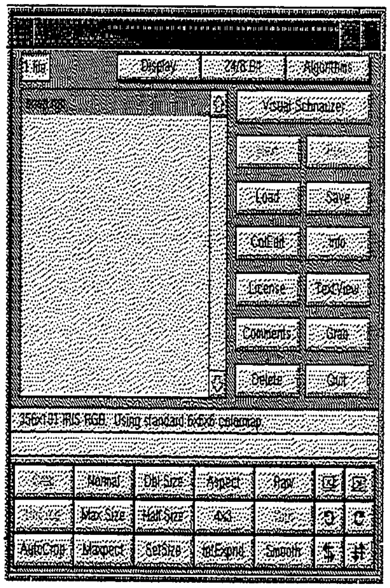


```
problem with impo
he object cannot b
y can be if a PS
```

```
% snapshot

prowler:/d4/herringt/
% snapshot
```

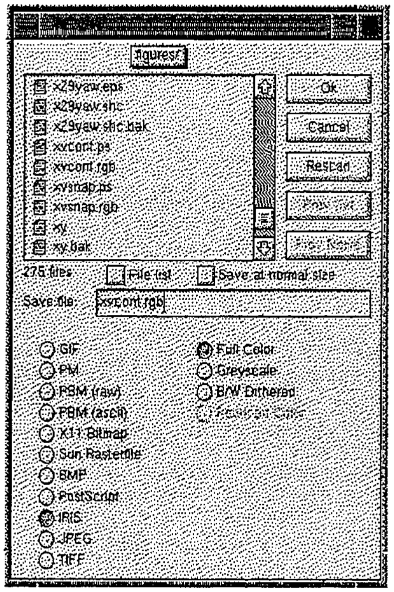
Figure A.12: XV Window

Place the mouse arrow on the picture and click the right mouse button. An "xv controls" window will appear, allowing the operator to save the file, figure A.13. Select the "Save" button on the "xv controls" window. A small fish will swim in circles until the "xv save" window appears, figure A.14. Select the "Postscript" button on the lower left side of the window, as well as the "B/W Dithered" button, if the file



356x191 IRIS RGB. Using standard 6x6x6 colormap

Figure A.13: XV Controls Window



x29yaw.eps
x29yaw.shc
x29yaw.shc.bak
xvcont.ps
xvcont.rgb
xvsnap.ps
xvsnap.rgb
xy
xy.bak

275 files □ File list □ Save at normal size

Save file: xvcont.rgb

○ GIF
○ PM
○ PBM (raw)
○ PBM (ascii)
○ X11 Bitmap
○ Sun Rasterfile
○ BMP
○ PostScript
● IRIS
○ JPEG
○ TIFF

● Full Color
○ Greyscale
○ B/W Dithered

Figure A.14: XV Save Window

is to be saved in black and white. Choose the file name to save and select the "Ok" button in the upper right corner. The file can now be accessed as a Postscript file and imported into any word processing program that accepts that format.

D. SHOWCASE PRESENTATIONS

Through the use of "Showcase" two animations have been developed to provide a visual depiction of the coordinate systems relationships and the coordinate transformation from navigation to body frames. These are provided on the following pages. They can be accessed through the "Showcase" browser window.

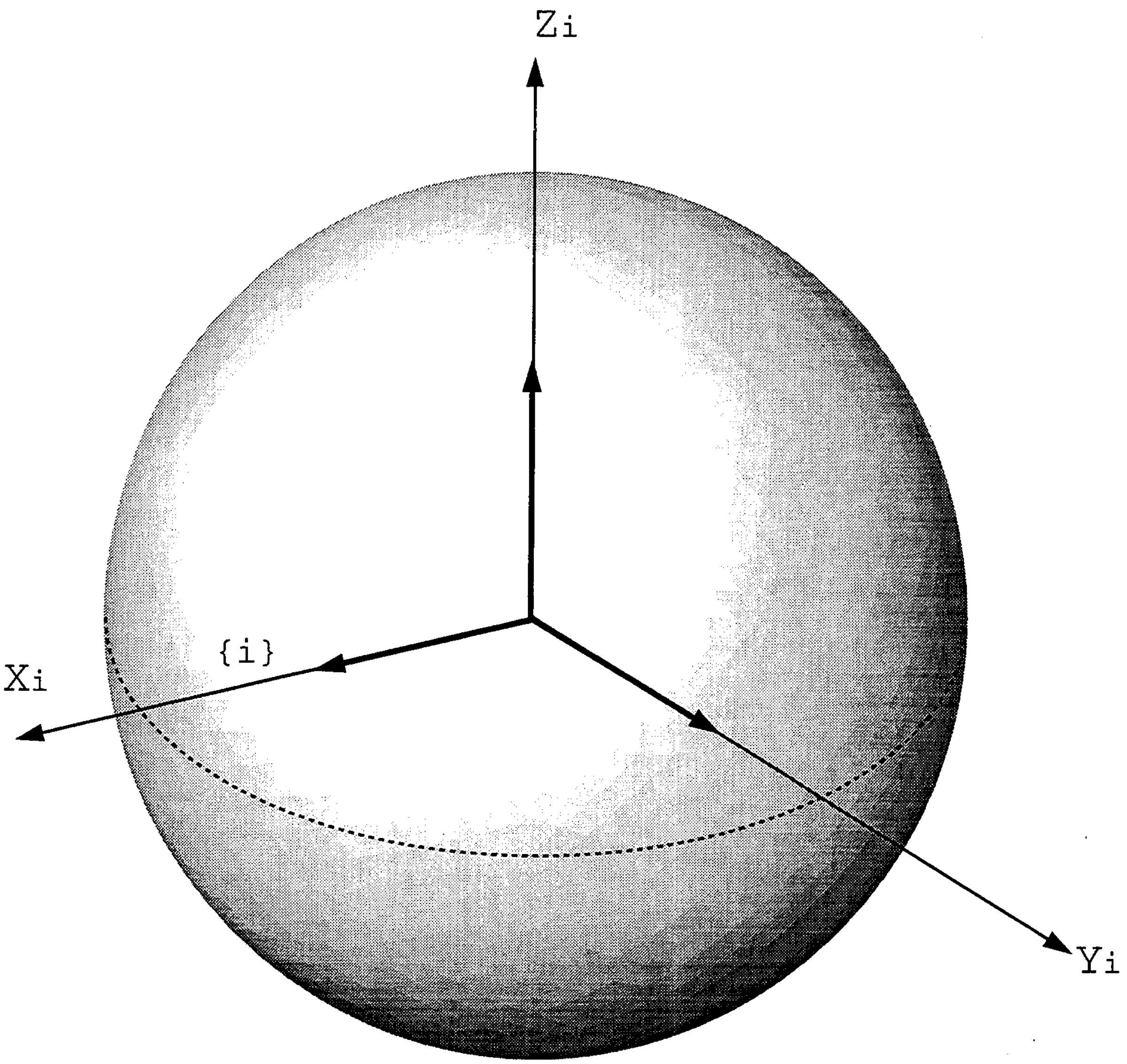


Figure A.15: Coordinate Transformation, Slide 1

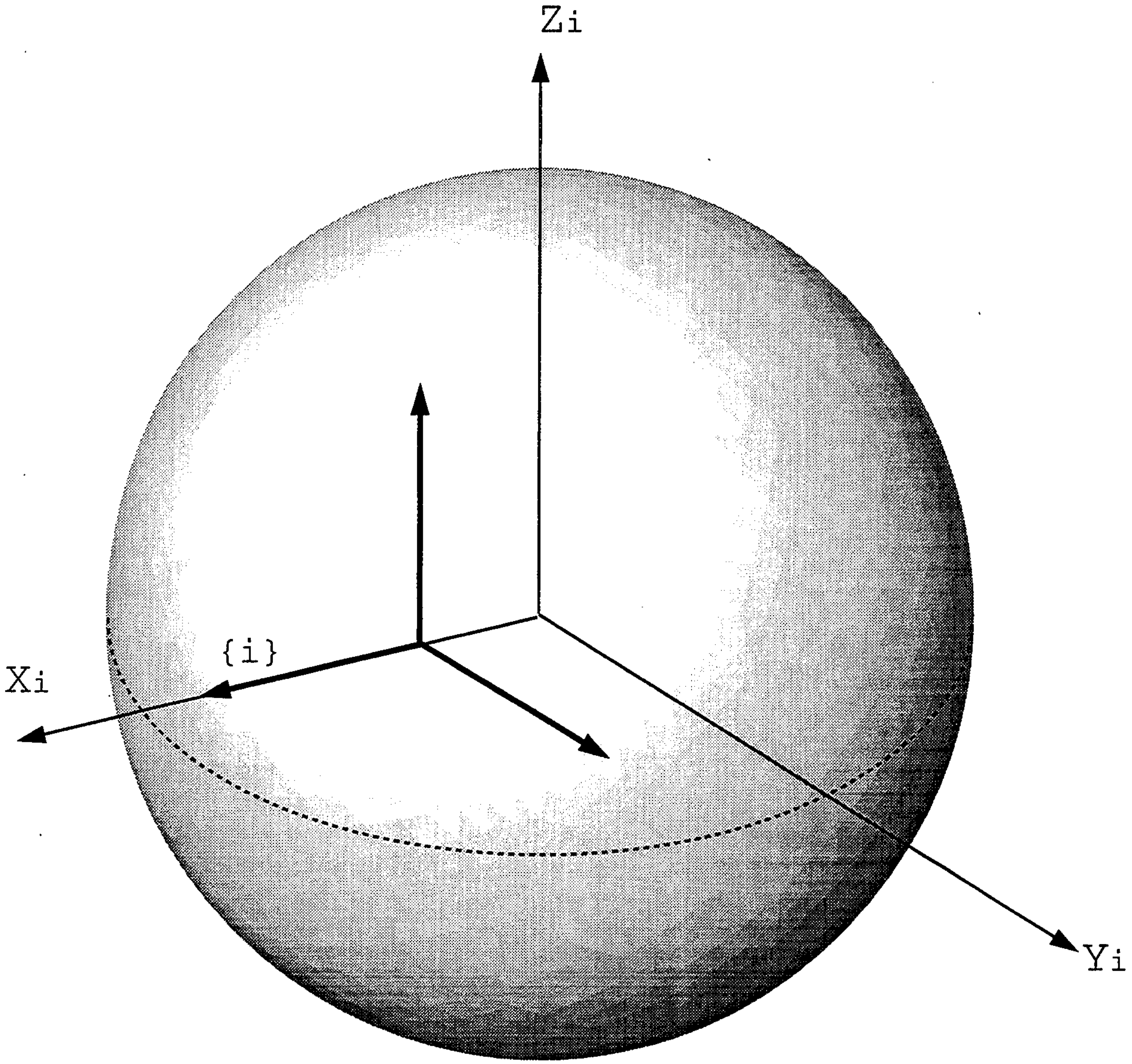


Figure A.16: Coordinate Transformation, Slide 2

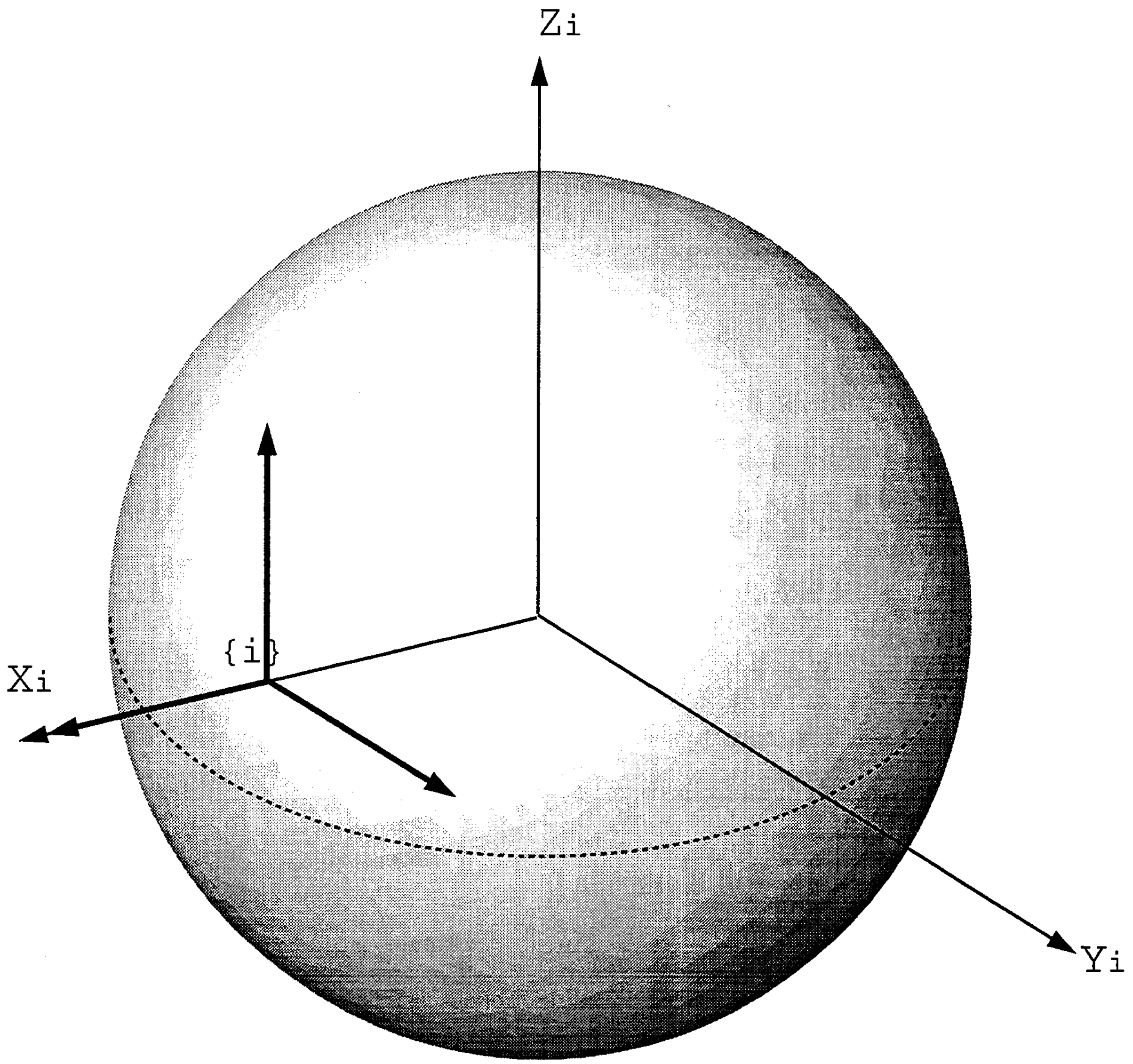


Figure A.17: Coordinate Transformation, Slide 3

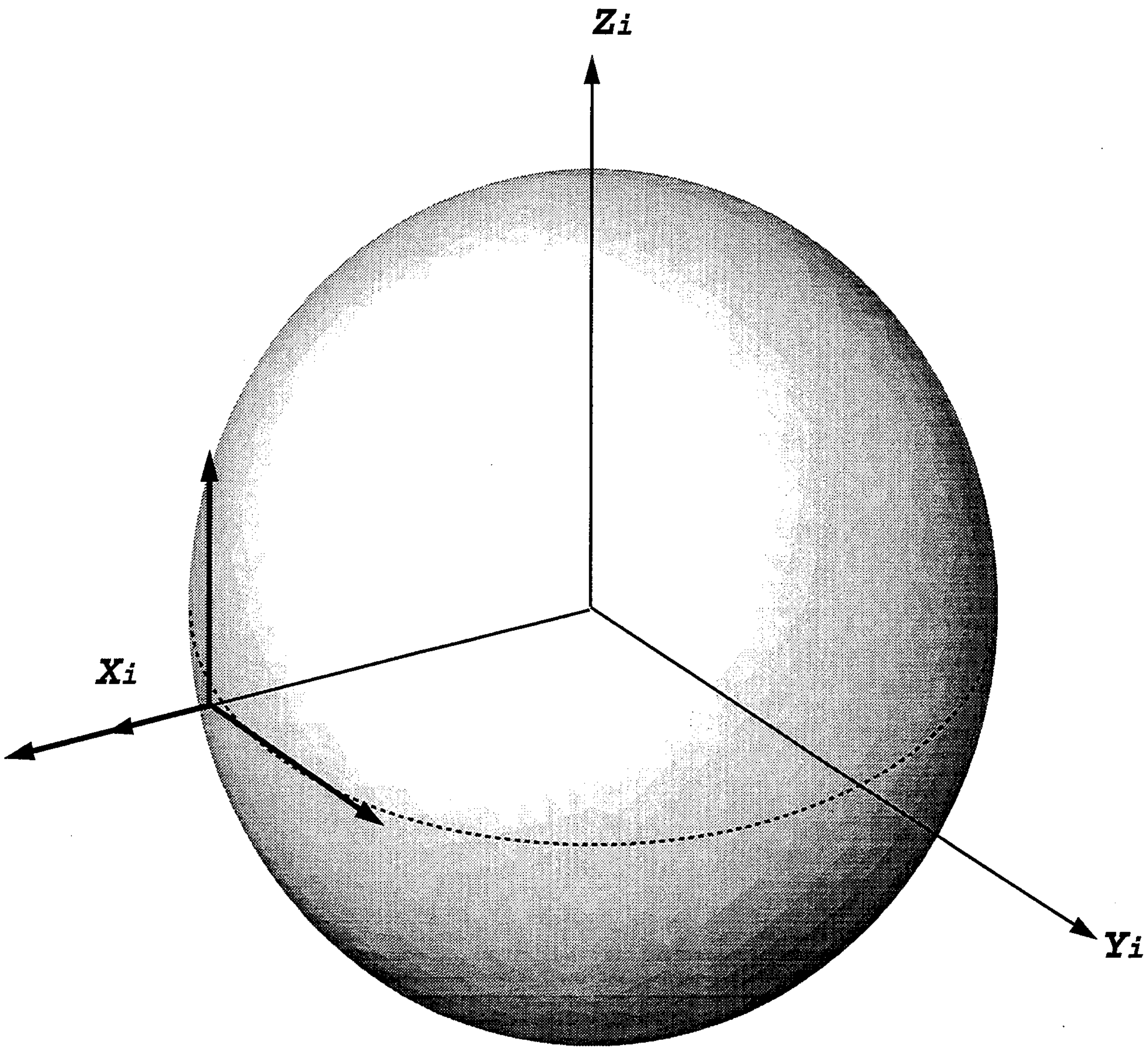


Figure A.18: Coordinate Transformation, Slide 4

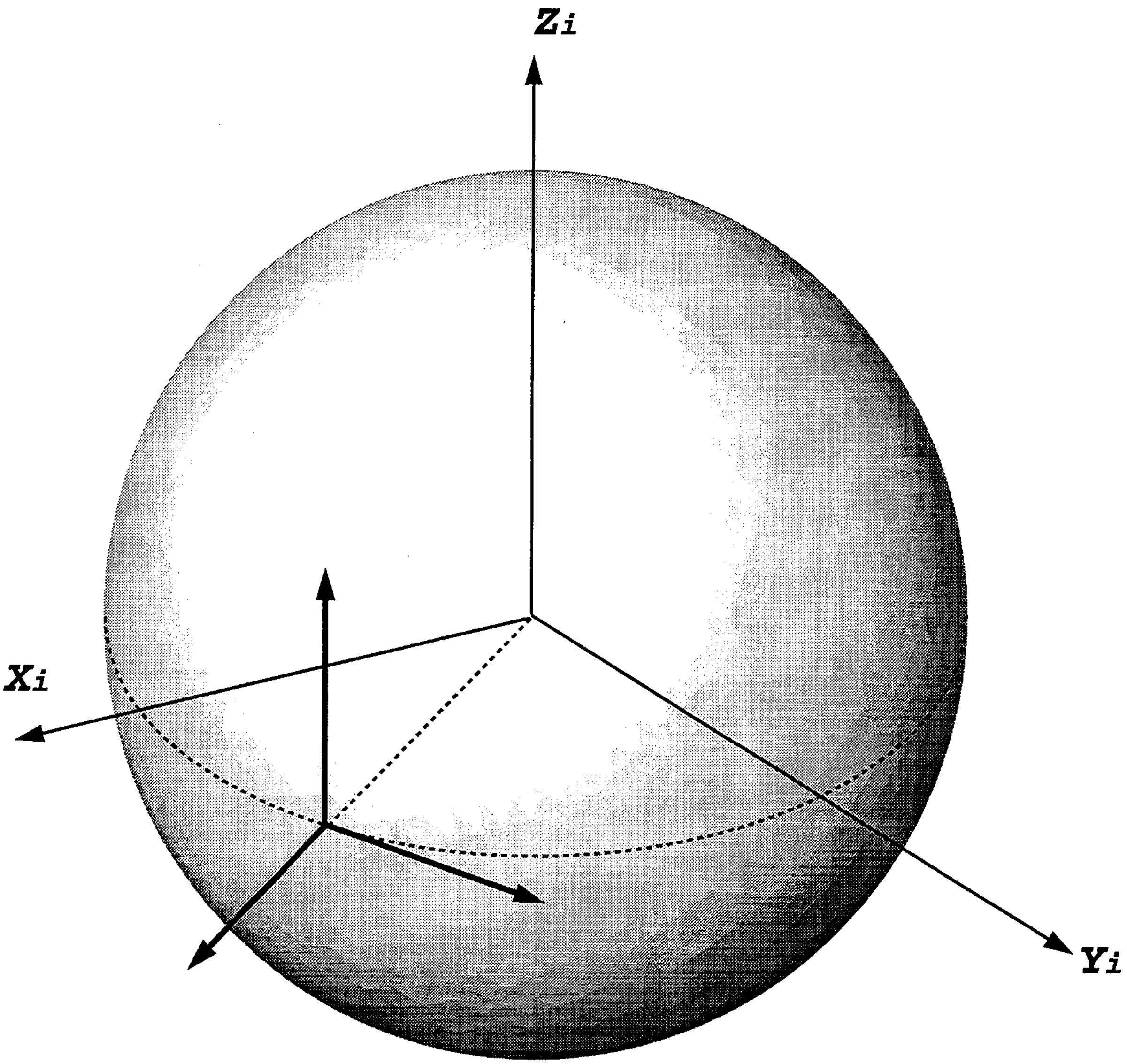


Figure A.19: Coordinate Transformation, Slide 5

Rotation about z_i

$$\begin{bmatrix} c\lambda & s\lambda & 0 \\ -s\lambda & c\lambda & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

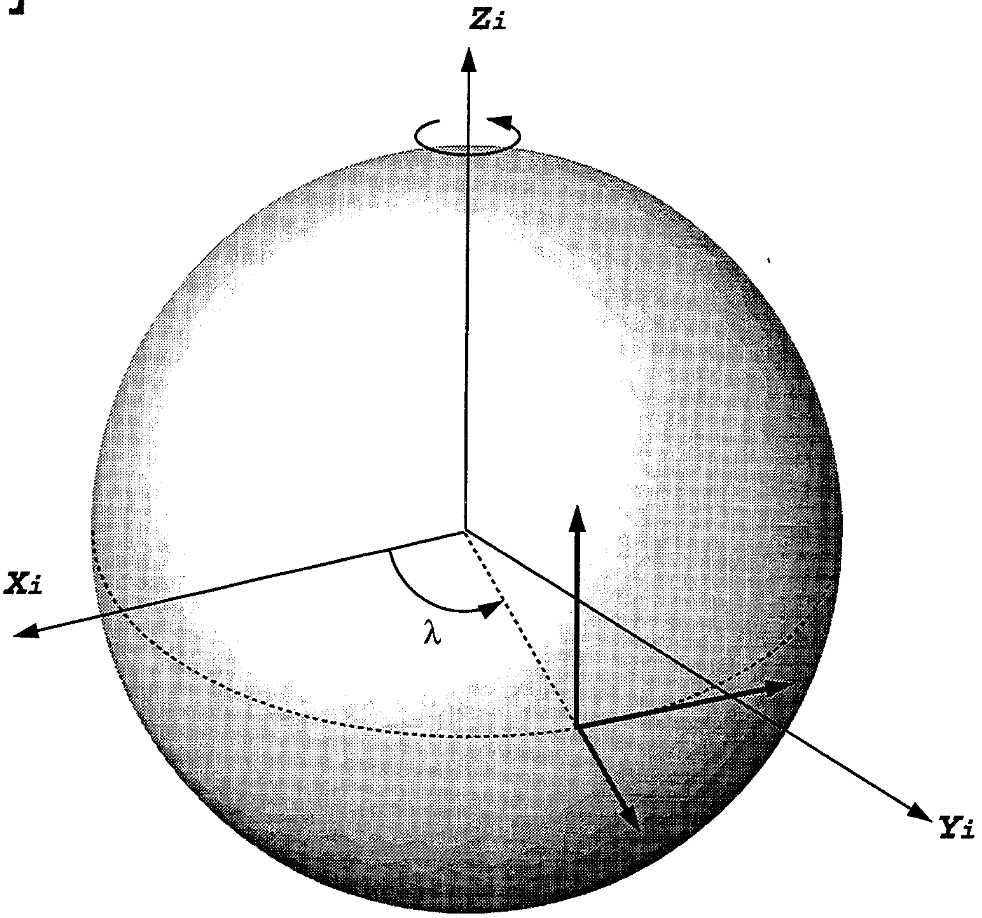


Figure A.20: Coordinate Transformation, Slide 6

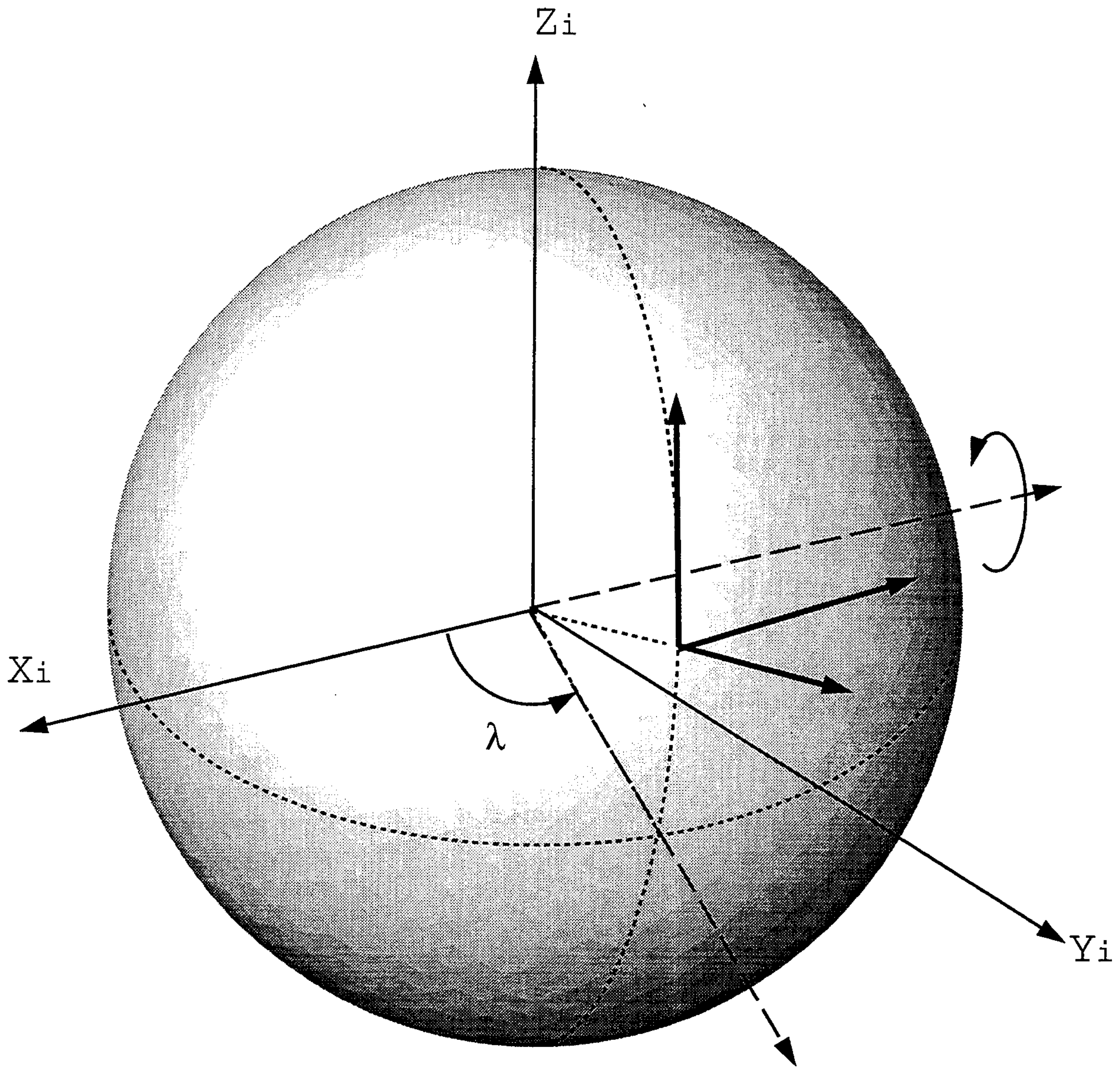


Figure A.21: Coordinate Transformation, Slide 7

Rotation about $Y_{i'}$

$$\begin{bmatrix} c\phi & 0 & s\phi \\ 0 & 1 & 0 \\ -s\phi & 0 & c\phi \end{bmatrix}$$

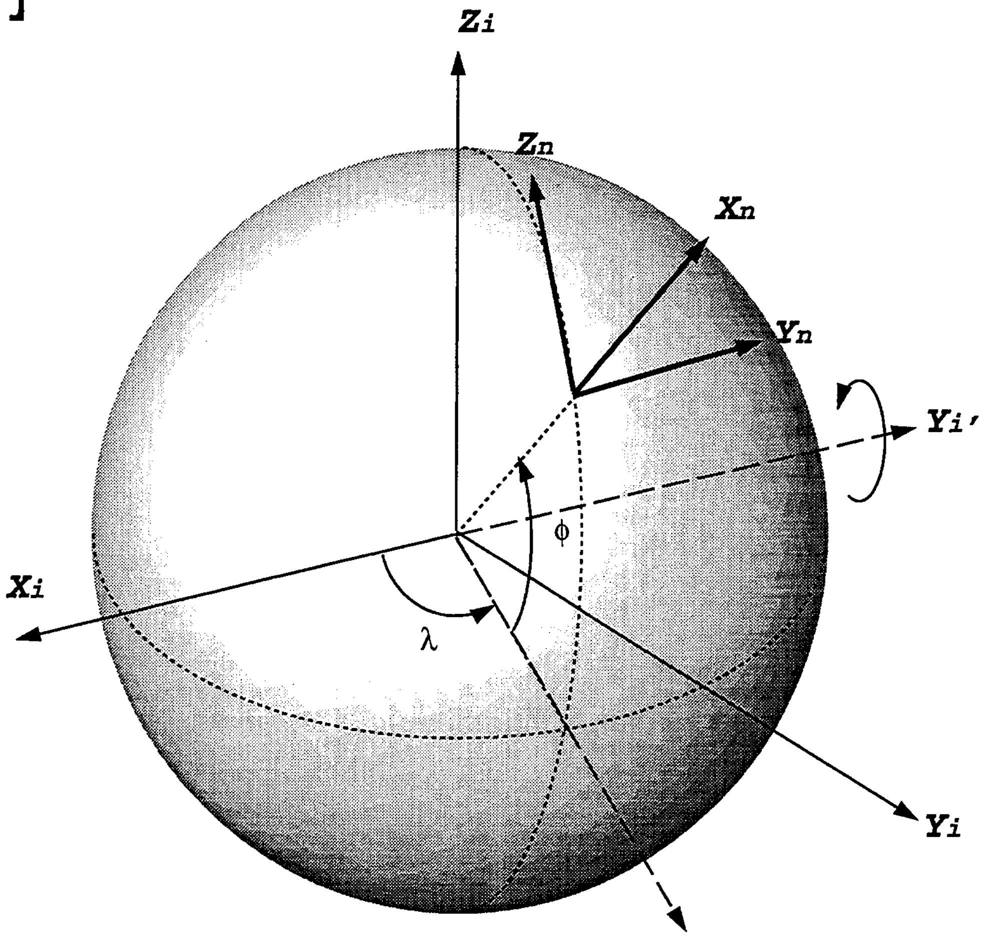


Figure A.22: Coordinate Transformation, Slide 8

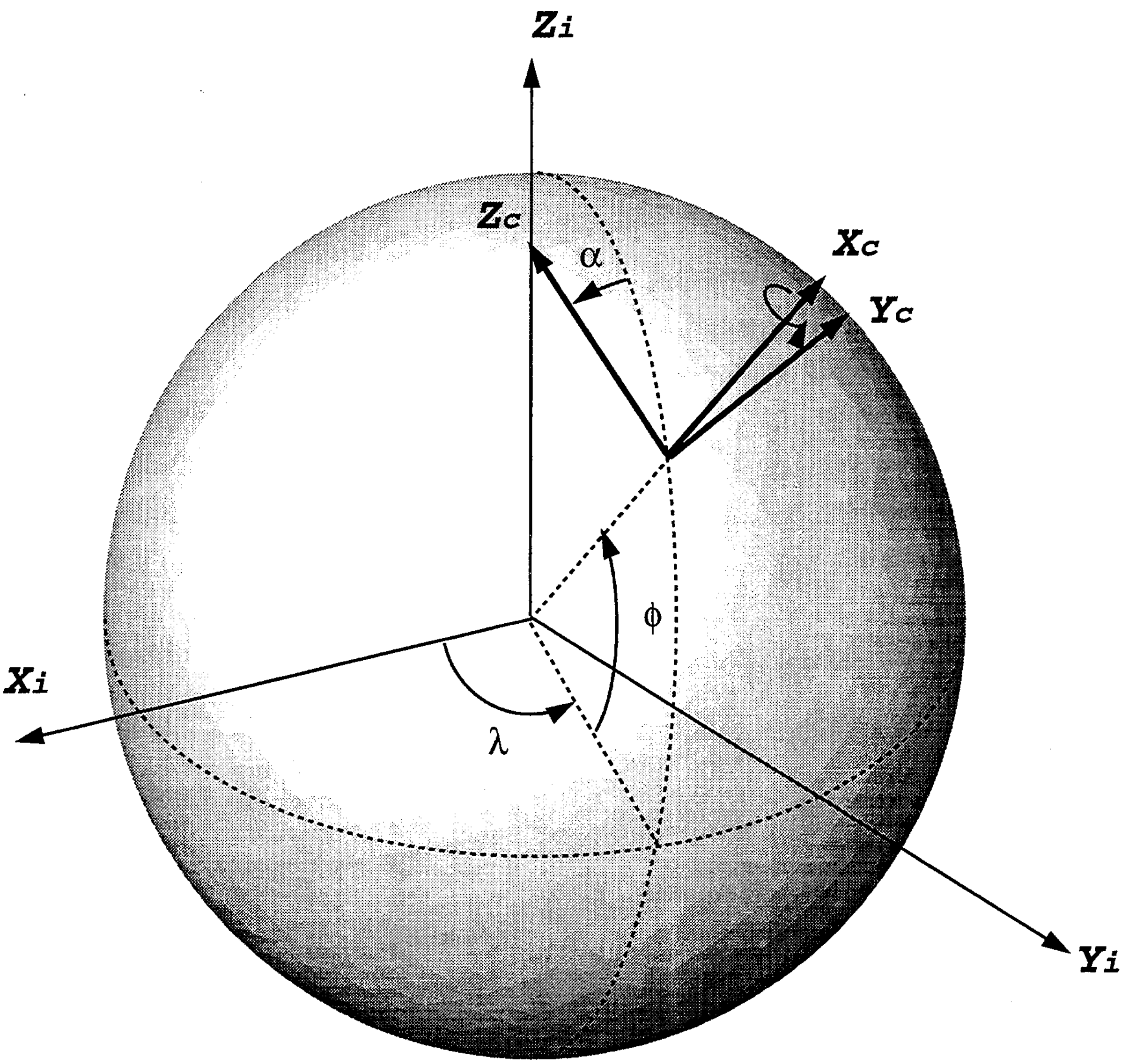
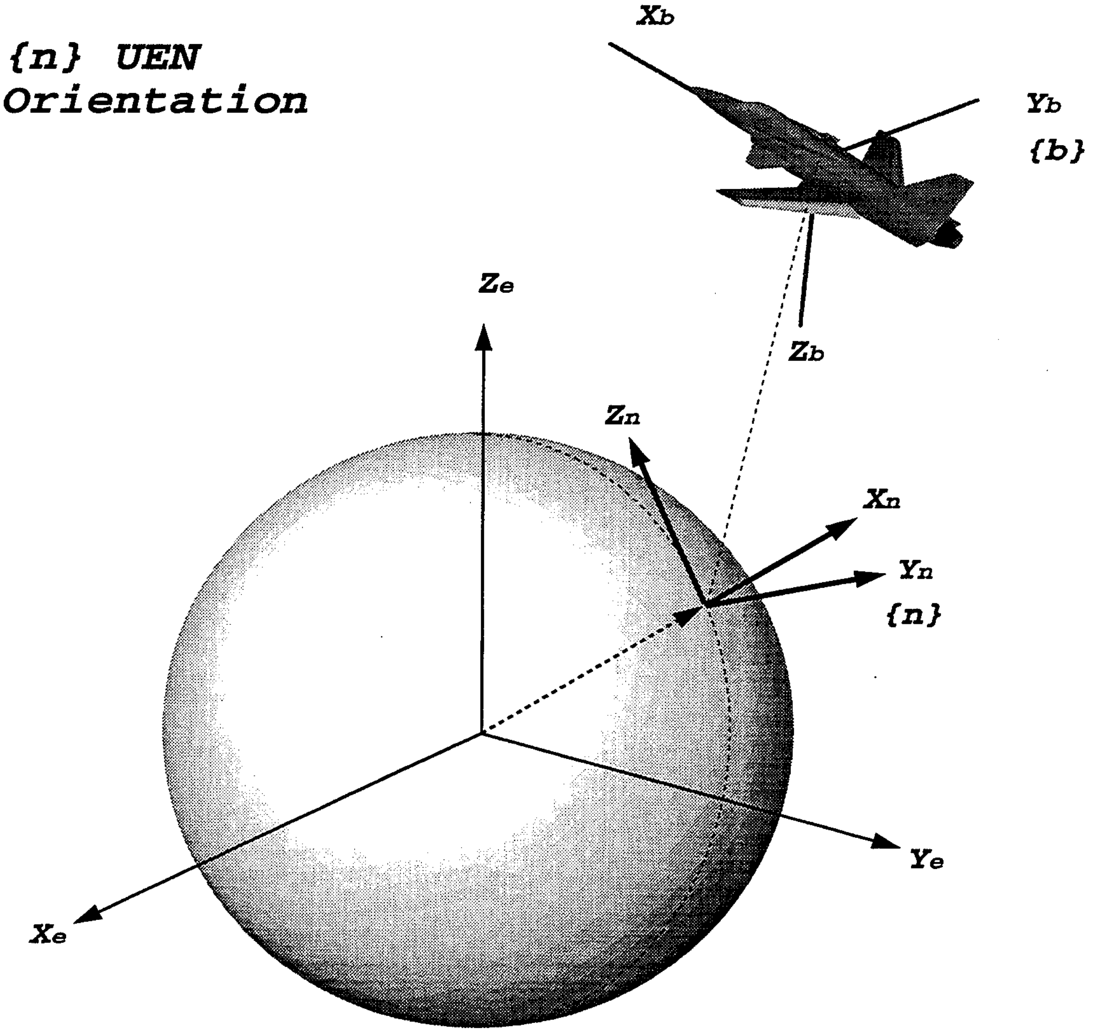


Figure A.23: Coordinate Transformation, Slide 9



{n} UEN Orientation

Figure A.24: Navigation to Body Rotation, Slide 1

{n} UEN Orientation

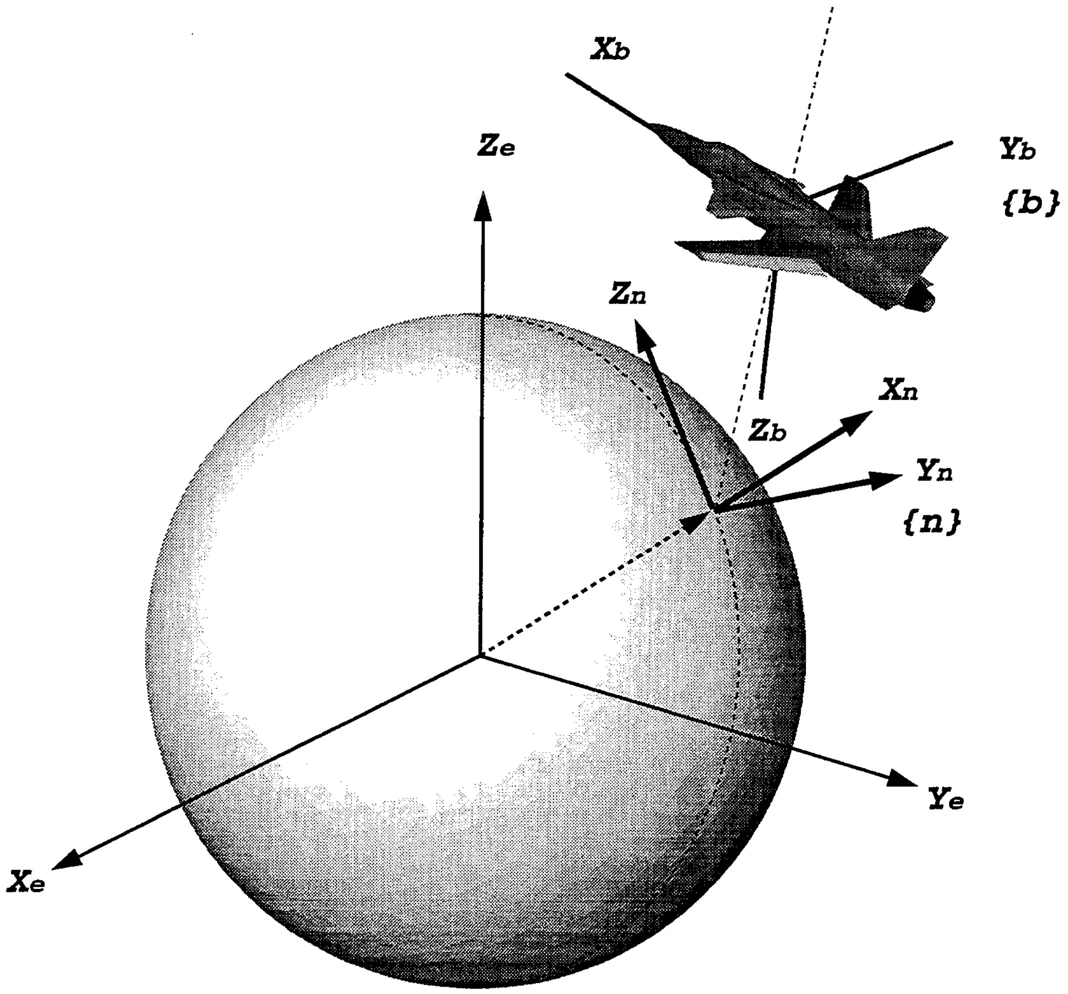


Figure A.25: Navigation to Body Rotation, Slide 2

{n} UEN
Orientation

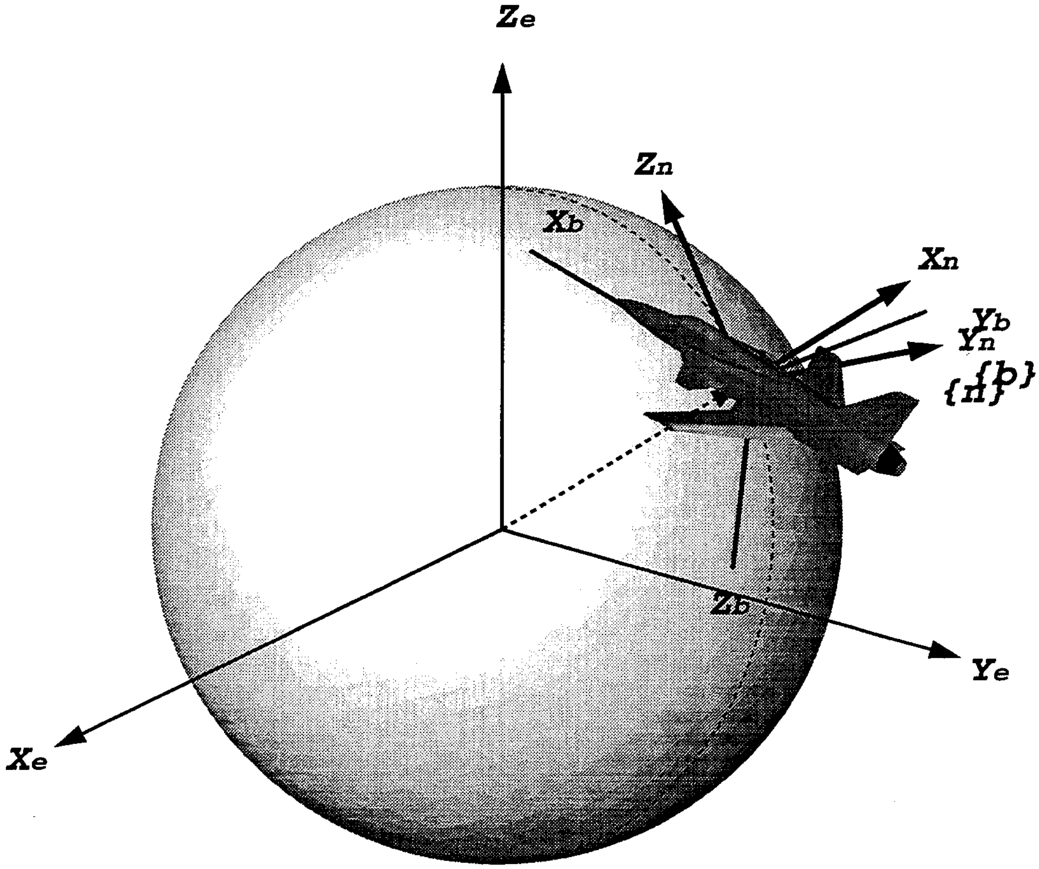


Figure A.26: Navigation to Body Rotation, Slide 3

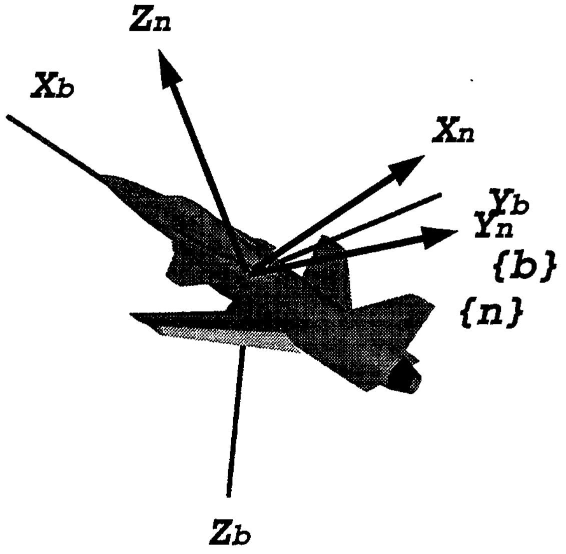


Figure A.27: Navigation to Body Rotation, Slide 4

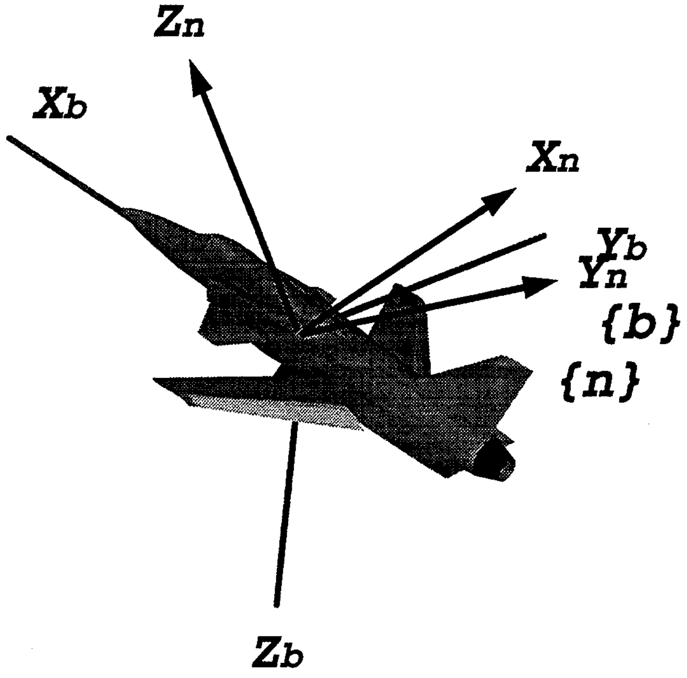


Figure A.28: Navigation to Body Rotation, Slide 5

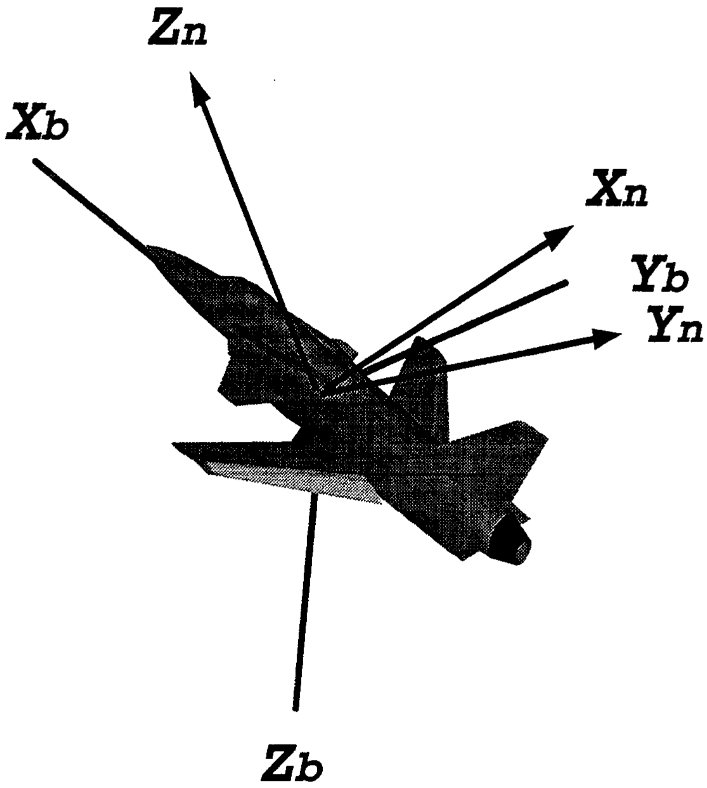


Figure A.29: Navigation to Body Rotation, Slide 6

Orient {n} from UEN
to NED. Rotate
90 degrees about Y_n

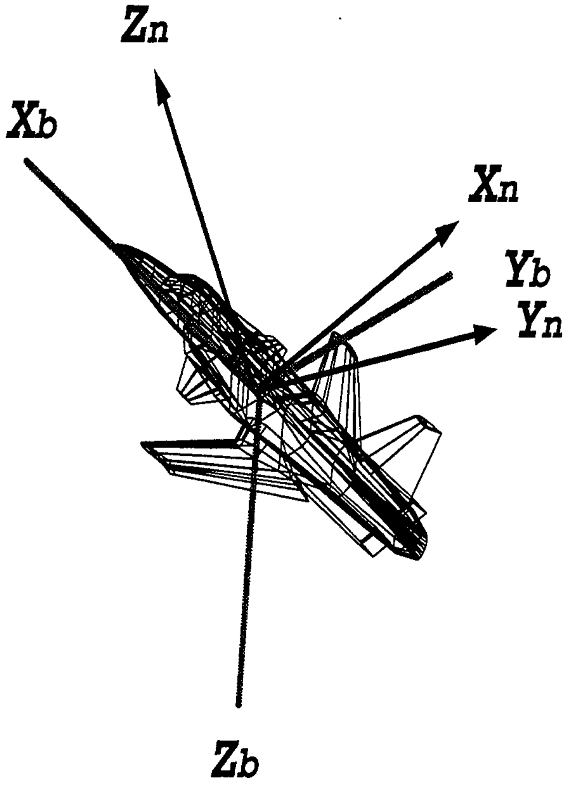


Figure A.30: Navigation to Body Rotation, Slide 7

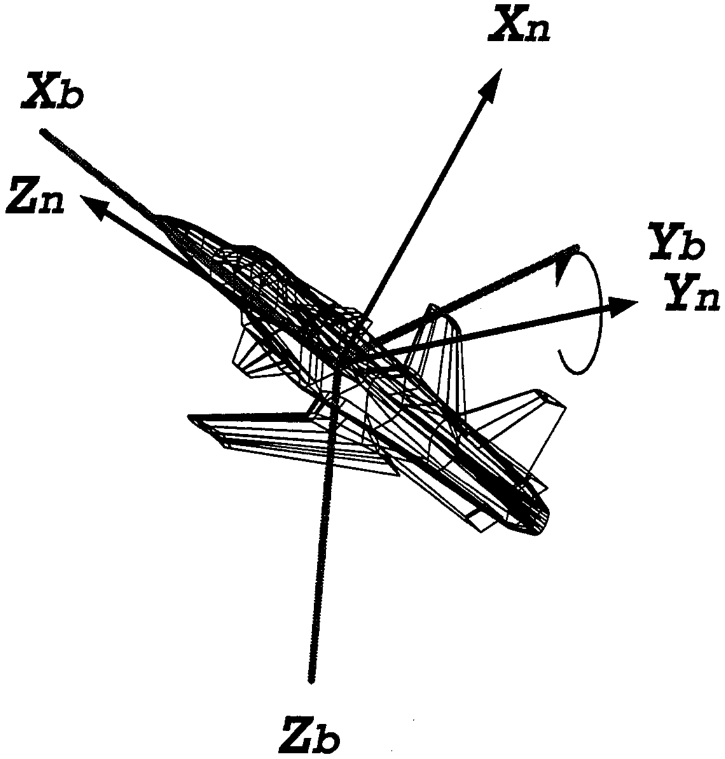


Figure A.31: Navigation to Body Rotation, Slide 8

$$\begin{bmatrix} \cos\left(\frac{\pi}{2}\right) & 0 & -\sin\left(\frac{\pi}{2}\right) \\ 0 & 1 & 0 \\ \sin\left(\frac{\pi}{2}\right) & 0 & \cos\left(\frac{\pi}{2}\right) \end{bmatrix}$$

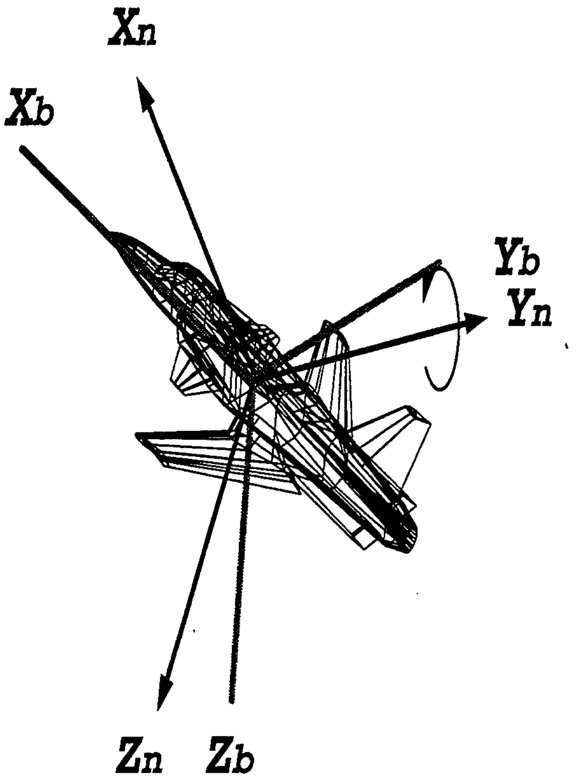


Figure A.32: Navigation to Body Rotation, Slide 9

Now rotate {n} about the Euler angles:

ψ, θ, ϕ

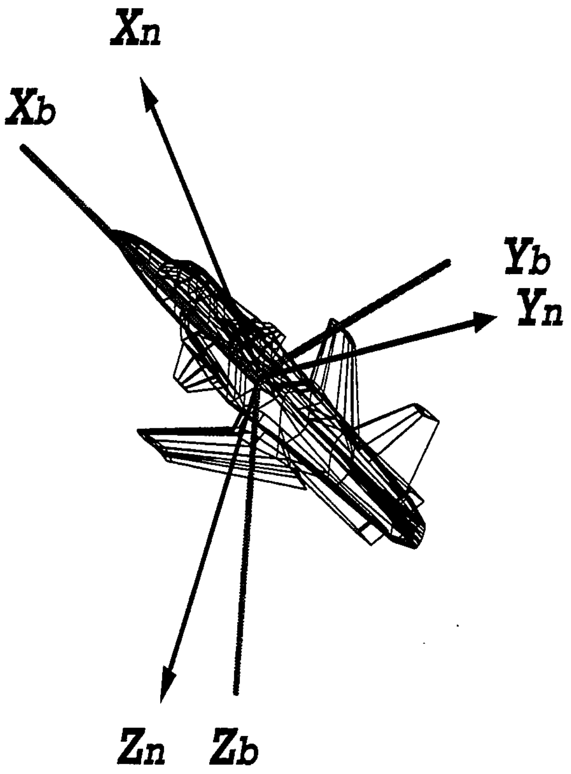


Figure A.33: Navigation to Body Rotation, Slide 10

Is this a positive or negative rotation? Does it matter?

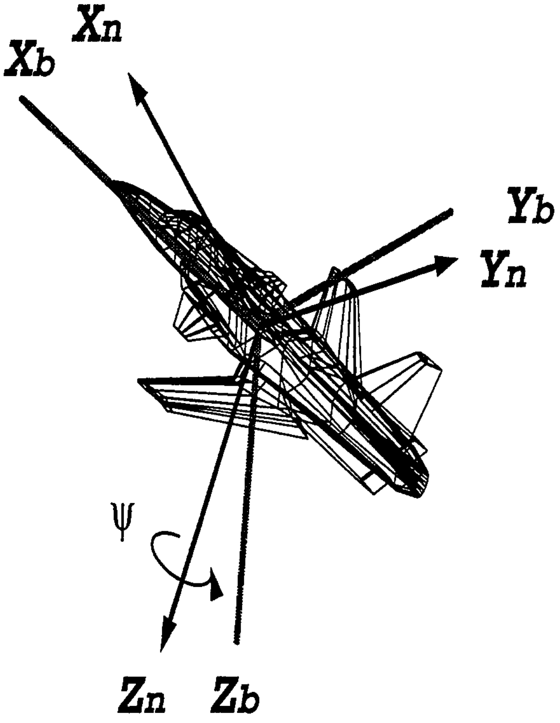


Figure A.34: Navigation to Body Rotation, Slide 11

$$\begin{bmatrix} \cos\psi & -\sin\psi & 0 \\ \sin\psi & \cos\psi & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

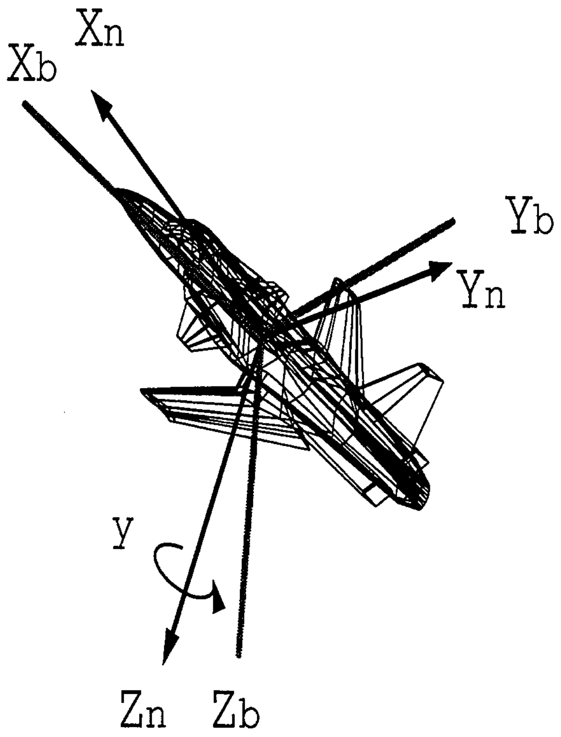


Figure A.35: Navigation to Body Rotation, Slide 12

Now rotate about Y_n. This aligns the x axes. Is this a positive rotation? Right hand rule!

$$\begin{bmatrix} \cos\theta & 0 & \sin\theta \\ 0 & 1 & 0 \\ -\sin\theta & 0 & \cos\theta \end{bmatrix}$$

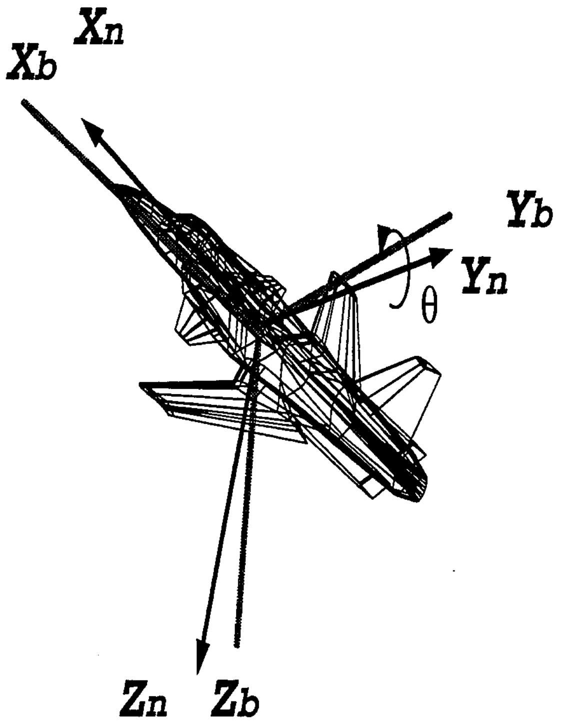


Figure A.36: Navigation to Body Rotation, Slide 13

Now rotate about the x_n axis. This will align the y and z axes.

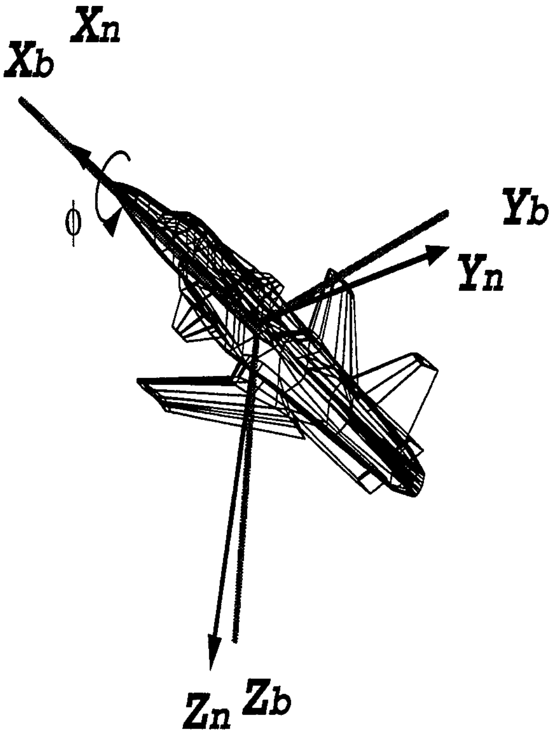


Figure A.37: Navigation to Body Rotation, Slide 14

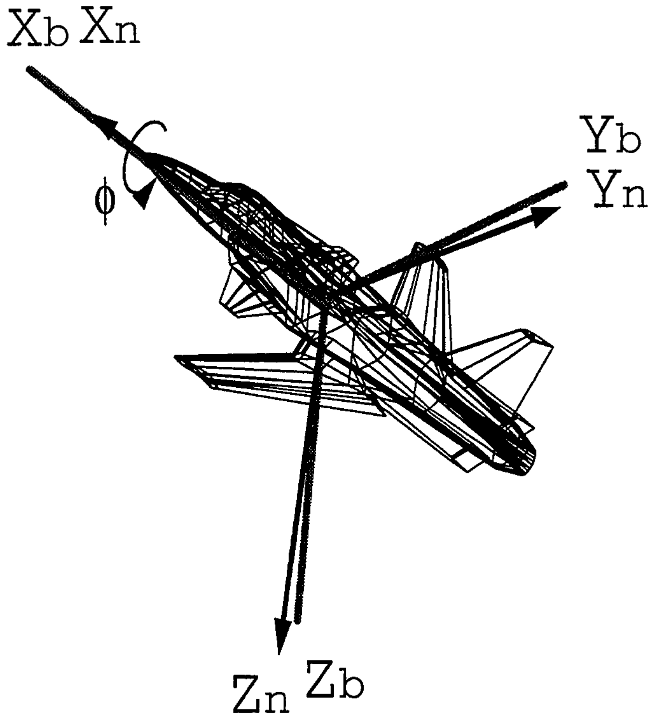


Figure A.38: Navigation to Body Rotation, Slide 15

$$\begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\phi & -\sin\phi \\ 0 & \sin\phi & \cos\phi \end{bmatrix}$$

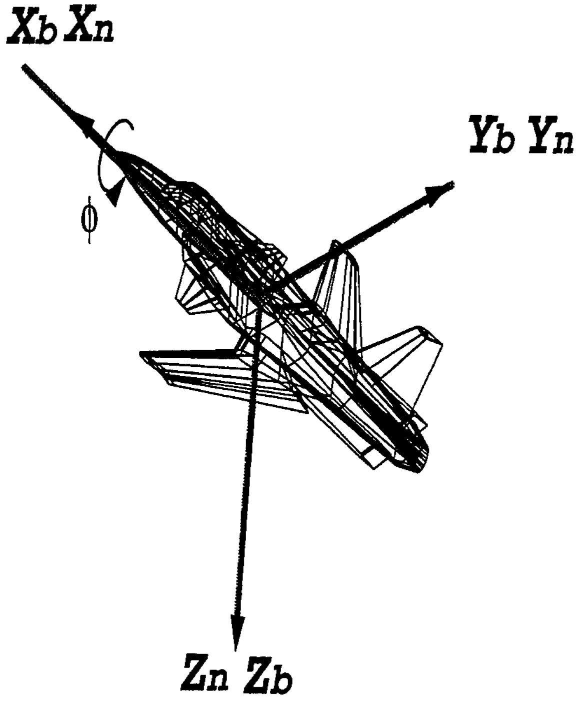


Figure A.39: Navigation to Body Rotation, Slide 16

APPENDIX B: MATLAB FUNCTIONS

```
**********************************************************************

MATLAB functions used in the IMU / GPS Integration

**********************************************************************

function y=abn2un(vec)

% Transform noisy accelerations from body fixed to universal


Ax=vec(1);

Ay=vec(2);

Az=vec(3);


euler=vec(4:6);


Tb2u=rb2u(euler);


y=Tb2u*vec(1:3);


**********************************************************************

function Ab=accel(vec)

% This function calculates the acceleration terms in the body

% coordinate frame


% Establish the order of the states

vbo=vec(1:3);
```

```
wb=vec(4:6);
Lambda=vec(7:9);
PboU=vec(10:12);
dvbo=vec(13:15);

% Calculate the cross product of wb and vbo
xprod=crpr(wb,vbo);

% Transform Lambda into the body fixed coordinate frame
BLambda=ru2b(Lambda);

% Multiply BLambda by the gravity term gu
gb=BLambda*gravitya(PboU);

% Calculate dvbo + (wb X vbo) - Ru2b*gu
Ab=dvbo+xprod-gb;

**********************************************************************
function y = crpr(omega,x)
% This subroutine computes the crossproduct of omega and x
% y = omega X x
p = omega(1);
q = omega(2);
r = omega(3);
t = [0 -r q; r 0 -p;-q p 0];
y = t * x;
```

```
**********************************************************************

function EOMstate=eom1(vec)


% This function will calculate the accelerations due to forces,

% moments, and control inputs.


% (1) Input aircraft data and initialize velocity terms:


Vi=677; % Initial velocity (fps)

W=13000; % Weight

S=230; % Wing planform area

b=34; % Wing span

cbar=7; % Mean aerodynamic chord

alpha_i=.047; % Initial angle of attack (rad) or 2.7 degrees

beta_i=0; % Initial sideslip angle

rho=.000588; % Density at 40,000 ft

g=32.174; % Gravitational constant

m=W/g; % Mass of the aircraft

To=1036.75; % Initial thrust

No=zeros(3,1);

Ib=[28000 0 -1300;0 18800 0;-1300 0 47000]; % Inertia tensor



Rw2b_i=[cos(alpha_i)*cos(beta_i),-cos(alpha_i)*sin(beta_i),sin(alpha_i);

-sin(beta_i),cos(beta_i),0;
```

```
sin(alpha_i)*cos(beta_i) -sin(alpha_i)*sin(beta_i) cos(alpha_i)];

[vbo]=Rw2b_i*[Vi 0 0]';
uo=vbo(1);vo=vbo(2);wo=vbo(3);

% (2) Input the derivative coefficients due to the states

dCx=[.104 0 .3 0 0 0;
      0 -.73 0 0 0 .4;
     .40 0 5.84 0 4.7 0;
      0 -.110 0 -.45 0 .16;
     .05 0 -.64 0 -15.5 0;
     0 .127 0 -.008 0 -.20];

dCxdot=[0 0 0 0 0 0;
0 0 0 0 0 0;
0 0 2.2 0 0 0;
0 0 0 0 0 0;
0 0 -6.7 0 0 0;
0 0 0 0 0 0];

dCdelt=[0 0 0;
0 0 .140;
.46 0 0;
0 .178 .019;
-1.24 0 0;
```

```
0 -.02 -.074];

Cf1=[.0335 0 .41 0 0 0]';

% (3) Define state variables
vb=vec(1:3);
u=vb(1); v=vb(2); w=vb(3);
wb=vec(4:6);  p=wb(1); q=wb(2); r=wb(3);
lambda=vec(7:9); phi=lambda(1); theta=lambda(2); psi=lambda(3);
cont=vec(10:12); elev=cont(1); ail=cont(2); rud=cont(3);
dt=vec(13);

% (4) Determine total velocity and dynamic pressure

V=sqrt(u^2 + v^2 + w^2);
qbar=.5*rho*V^2;

% (5) Determine angle of attack (alpha) and sideslip (beta)

alpha=asin(w/V);
beta=asin(v/V);

% (6) Develop diagonal Sbar matrix

Sbar=diag([-S,S,-S,S*b,S*cbar,S*b]);
```

```
% (7) Develop M1 and M2 matrices

M1=diag([1/V,1/V,1/V,b/(2*V),cbar/(2*V),b/(2*V)]);
M2=diag([0,.5*b/V^2,.5*cbar/V^2,0,0,0]);

% (8) Calculate mass inertia matrix

MI=[eye(3)/m zeros(3,3);zeros(3,3) Ib\eye(3)];
% (9) Develop wind to body transformation matrix

Rw2b=[cos(alpha)*cos(beta) -cos(alpha)*sin(beta) -sin(alpha);
sin(beta) cos(beta) 0;
sin(alpha)*cos(beta) -sin(alpha)*sin(beta) cos(alpha)];

Tw2b=[Rw2b zeros(3,3); zeros(3,3) Rw2b];

% (10) Calculate linear and angular velocity terms

SW=-crpr(wb,vb);
ISIW=(Ib)\crpr(Ib*wb,wb);
SWM=[SW ; ISIW];
Y=MI*Tw2b.*qbar*Sbar*dCx*M1*[vb-vbo;wb];

% (11) Calculate universal to body rotation matrix (3-2-1)

Rpsi=[cos(psi) sin(psi) 0; -sin(psi) cos(psi) 0; 0 0 1];
```

```
Rtheta=[cos(theta),0,-sin(theta);0,1,0;sin(theta),0,cos(theta)];
Rphi=[1 0 0; 0 cos(phi) sin(phi); 0 -sin(phi) cos(phi)];

Ru2b=Rphi*Rtheta*Rpsi;
Rb2u=Ru2b';
% (12) Calculate gravity and control terms

Fgravu=[0 0 g]';

Fgravb=[Ru2b*Fgravu; zeros(3,1)];

% Calculate thrust force terms.  Assume thrust acts through the cg
% and there are no moments due to thrust

T=[To/m;0;0];
Fthrust=[T;(Ib)\No]*dt;

% (13) Calculate aero forces and moments

FaeroM=MI*Tw2b.*qbar*Sbar;

FaeroC=[Cf1+dCxdot*M2*[(crpr(wb,vb-vbo));zeros(3,1)]+dCdelt*cont];
Faero=FaeroM*FaeroC;

% (14) Calculate chi
```

```
chi=[eye(6) - MI*Tw2b.*qbar*Sbar*dCxdot*M2];

% (15) Calculate accelerations

EOM1_6 = chi\(SWM + Y + (Fgravb + Fthrust + Faero));

% (16) Calculate Lambdadot

QLambda=[1 sin(phi)*tan(theta) cos(phi)*tan(theta);
0 cos(phi) -sin(phi);
0 sin(phi)*sec(theta) cos(phi)*sec(theta)];

EOM7_9=QLambda*wb;  % Lambdadot


% (19) Calculate state output

EOMstate=[EOM1_6;EOM7_9];

***********************************************************************
function xout = exact(vec)
% This function computes the pseudo ranges by adding the clock
% bias range to the actual range.  Input vector consists of the
% actual aircraft position (PboU), the position of each satellite
% (Psi), euler angles, user equivalent range error (re)
```

```
Ro=20.997e6;

Puoe=[Ro 0 0]'; % Position of the tangent plane

% (Greenwich meridian)

% Assume tangent plane origin remains at

% the Greenwich Meridian.


Pbou=vec(1:3); % True aircraft positions from EOM


Pos1=vec(4:6); % Satellite positions in earth fixed frame

Pos2=vec(7:9);

Pos3=vec(10:12);

Pos4=vec(13:15);



%Ru2e=[0 0 1; % Transform tangent plane to earth fixed

%0 1 0;

%1 0 0];


Pboe=Pbou + Puoe; % Transform aircraft position from body to

% earth fixed



rho1=norm(Pboe+Puoe-Pos1);

rho2=norm(Pboe+Puoe-Pos2);

rho3=norm(Pboe+Puoe-Pos3);

rho4=norm(Pboe+Puoe-Pos4);
```

```
xout=[rho1 rho2 rho3 rho4]';

**********************************************************************
function g = gravitya(r)
%
% This function computes the gravity vector in the tangent plane
% r is the vector of the vehicle's position in tangent plane
%
R0 = 20.997e6;
mu = 32.2*R0^2;
R = (r(1)^2 + r(2)^2 + (r(3) + R0)^2)^.5;
gx = -(mu/R^3)*r(1);
gy = -(mu/R^3)*r(2);
gz = -(r(3) + R0)*mu/R^3;
g = [gx; gy; gz];

**********************************************************************
function satpos=initsat
% Initialize satellite and GPS receiver positions


Rs=285.885e6; c1=cos(2*pi/3); s1=sin(2*pi/3); s2=-s1;
satpos1=[0  0  Rs]'; satpos2=[Rs  0  0]';
satpos3=[Rs*c1 Rs*s1 0]';  satpos4=[Rs*c1 Rs*s2  0]';
satpos=[satpos1' satpos2' satpos3' satpos4'];
```

```
**********************************************************************

function out=pos(vec)

% This function converts velocity in the body frame to velocity in

% the inertial frame


vb=vec(1:3); u=vb(1); v=vb(2); w=vb(3);

wb=vec(4:6);  p=wb(1); q=wb(2); r=wb(3);

lambda=vec(7:9); phi=lambda(1); theta=lambda(2); psi=lambda(3);

Rpsi=[cos(psi) sin(psi) 0; -sin(psi) cos(psi) 0; 0 0 1];

Rtheta=[cos(theta) 0 -sin(theta); 0 1 0; sin(theta) 0 cos(theta)];

Rphi=[1 0 0; 0 cos(phi) sin(phi); 0 -sin(phi) cos(phi)];


Ru2b=Rphi*Rtheta*Rpsi;


Rb2u=Ru2b';


out=Rb2u*vb;


**********************************************************************

function xout = pseudo(vec)

% This function computes the pseudo ranges by adding the clock

% bias range to the actual range.  Input vector consists of the

% actual aircraft position (PboU), the position of each satellite

% (Psi), euler angles user equivalent range error (re)
```

```
Ro=6.38e6;
Puoe=[Ro 0 0]'; % Position of the tangent plane.
% Assume tangent plane origin remains at
% the Greenwich Meridian.

Pbou=vec(1:3); % True aircraft positions from EOM

Pos1=vec(4:6); % Satellite positions in earth fixed frame
Pos2=vec(7:9);
Pos3=vec(10:12);
Pos4=vec(13:15);

re=vec(16); % User equivalent range error

Ru2e=[0 0 1; % Transform tangent plane to earth fixed
0 1 0;
1 0 0];

Pboe=Ru2e*Pbou + Puoe; % Transform aircraft position from body to
% earth fixed


rho1=norm(Pboe+Puoe-Pos1)+ re;
rho2=norm(Pboe+Puoe-Pos2)+ re;
rho3=norm(Pboe+Puoe-Pos3)+ re;
rho4=norm(Pboe+Puoe-Pos4)+ re;
```

```
xout=[rho1 rho2 rho3 rho4]';

***********************************************************************
function y=rb2u(x)

% Calculate inertial to body rotation matrix (3-2-1)


phi=x(1);
theta=x(2);
psi=x(3);

Rpsi=[cos(psi) sin(psi) 0; -sin(psi) cos(psi) 0; 0 0 1];
Rtheta=[cos(theta) 0 -sin(theta); 0 1 0; sin(theta) 0 cos(theta)];
Rphi=[1 0 0; 0 cos(phi) sin(phi); 0 -sin(phi) cos(phi)];

y=[Rphi*Rtheta*Rpsi]';

***********************************************************************
function y=ru2b(x)

% Calculate body to inertial rotation matrix

phi = x(1); theta = x(2); psi = x(3);
Rpsi=[cos(psi) sin(psi) 0; -sin(psi) cos(psi) 0; 0 0 1];
```

```
Rtheta=[cos(theta) 0 -sin(theta); 0 1 0; sin(theta) 0 cos(theta)];
Rphi=[1 0 0; 0 cos(phi) sin(phi); 0 -sin(phi) cos(phi)];

y=Rphi*Rtheta*Rpsi;

*********************************************************************
% Function kal.m
% This function performs the linearization of the Kalman filter.

% First I will linearize the navigation model "navmod.m". The inputs
% are Ax, Ay, and Az from the IMU and drift error for the exact range
% calculations.  Outputs are the four pseudo ranges.

[a,b,c,d]=linmod('navmod')

pause

% Next, I will include these matrices in the synthesis model,
% "syn.m". Inputs 1 through 4 will be noisy accelerations where
% I will append transmissions zeros at -1 and include an integrator.
% Input 4 will be noise from the clock model.  Inputs 5 through 7
% will be the accelerations from the IMU.

[as,bs,cs,ds]=linmod('syn')

pause
```

```
% Next I will determine the Kalman gains from the noise inputs, 1-4.
% I chose Q=1 and R=1 initially.
Q=eye(4);R=eye(4);

l=lqe(as,bs(:,1:4),cs,Q,R)

% I will check the bode response of the pseudo range inputs to the
% pseudo range estimates. I will use a function called "bplots.m".

bplots

% I will adjust the weighting values to obtain a bandwidth of
% 1 rad/sec.
% Decrease R to 5.

l=lqe(as,bs(:,1:4),cs,Q,5*R)
bplots

l=lqe(as,bs(:,1:4),cs,.5*Q,5*R)
bplots

l=lqe(as,bs(:,1:4),cs,.05*Q,50*R)
bplots
```

```
l=lqe(as,bs(:,1:4),cs,.05*Q,6*R)
bplots

% I will use the values of Q=.05 and R=6.  These provide bandwidths
% of 1 rad/sec.

% Next, I will look at the response of the filter from the
% accelerations (input) to the pseudo ranges (output).  I will use
% a function called "accplot.m".

accplot

**********************************************************************

% Function bplots.m
% This function plots each input bode response

w=logspace(-4,2,100);
x=-3*ones(size(w));
[mag,phase]=bode(as-l*cs,l,cs,ds(:,1:4),1,w);
magdb=20*log(mag);
subplot(211)
semilogx(w,magdb,w,x,'.')
grid
title('Bode Response for Pseudo ranges 1 & 2')
```

```
axis([1e-4 1e2 -10 10])
pause
% print

[mag,phase]=bode(as-l*cs,l,cs,ds(:,1:4),2,w);
magdb=20*log(mag);
subplot(212)
semilogx(w,magdb,w,x,'.')
grid
axis([1e-4 1e2 -10 10])
pause
% print

[mag,phase]=bode(as-l*cs,l,cs,ds(:,1:4),3,w);
magdb=20*log(mag);
subplot(211)
semilogx(w,magdb,w,x,'.')
grid
title('Bode Response for Pseudo ranges 3 & 4')
axis([1e-4 1e2 -10 10])
pause
% print

[mag,phase]=bode(as-l*cs,l,cs,ds(:,1:4),4,w);
magdb=20*log(mag);
subplot(212)
```

```
semilogx(w,magdb,w,x,'.')
grid
axis([1e-4 1e2 -10 10])
pause
% print

**********************************************************************
% Function accplot.m
% This function plots each input bode response for the accelerations

w=logspace(-4,2,100);
x=-3*ones(size(w));
[mag,phase]=bode(as-l*cs,bs(:,5:7),cs,ds(:,5:7),1,w);
magdb=20*log(mag);
subplot(311)
semilogx(w,magdb,w,x,'.')
grid
title('Bode Response for Ax, Ay, Az')
axis([1e-4 1e2 -100 100])
pause
% print

[mag,phase]=bode(as-l*cs,bs(:,5:7),cs,ds(:,5:7),2,w);
magdb=20*log(mag);
subplot(312)
semilogx(w,magdb,w,x,'.')
```

```
grid


axis([1e-4 1e2 -100 100])

pause

% print


[mag,phase]=bode(as-l*cs,bs(:,5:7),cs,ds(:,5:7),3,w);

magdb=20*log(mag);

subplot(313)

semilogx(w,magdb,w,x,’.’)

grid


axis([1e-4 1e2 -100 100])

pause

% print


**********************************************************************

% Function plotstep.m

% This function linearizes the Kalman filter with gains and produces a

% step response due to accelerations in the x,y, and z axes.


[af,bf,cf,df]=linmod(’kfilnav2’);


[Y,X,t]=step(af,bf,cf,df);


plot(t,Y)
```

```
title('Step response due to inputs in Ax,Ay,Az')

pause
```

LIST OF REFERENCES

1. Siouris, G., *Aerospace Avionics Systems: a modern synthesis*, Academic Press, 1993.

2. Britting, K., *Inertial Navigation Systems Analysis*, John Wiley and Sons, 1971.

3. Craig, J., *Introduction to robotics: mechanics and control*, Addison–Wesley, 1989.

4. Roskam, J., *Airplane Flight Dynamics and Automatic Flight Controls*, Roskam Aviation and Engineering, 1979.

5. Marquis, C., "Design of a GPS Aided Guidance, Navigation, and Control System for Trajectory Control of an Air Vehicle," Master's Thesis, Department Of Aeronautics, Naval Postgraduate School, Monterey, CA, March 1994.

6. Kaminer, I., *Class Notes for AA4276,* U.S. Naval Postgraduate School, Monterey, CA. 1994.

7. Department of Defense and Department of Transportation, DOD–4650.4/DOT–TSC–RSPA–88–4, *1990 Federal Radionavigation Plan*, Appendix A, 1990.

8. Logsdon, T., *The NAVSTAR Global Positioning System*, Van Nostrand Reinhold, 1992.

9. NATO Team, NAVSTAR GPS Joint Program Office, *Introduction to NAVSTAR GPS User Equipment*, February, 1991.

10. "GPS Seminar", presented to the Naval Security Group at the Applied Research Laboratories of the University of Texas (Austin), January 26-28, 1993, vol. I and II.

11. Wells, D. et al., *Guide to GPS Positioning*, Canadian GPS Associates, 1987.

12. Feess, W., and Stephens, S., "Evaluation of GPS Ionospheric Time Delay Algorithm for Single Frequency Users", *Proceedings of the IEEE Position, Location and Navigation Symposium (PLANS '86)*, Las Vegas, NV, pp. 206-213, November 4–7, 1986.

13. Leick, A., *GPS Satellite Surveying*, John Wiley and Sons, 1990.

14. Black, H., "An Easily Implemented Algorithm for the Tropospheric Range Correction", Journal of Geophysical Research, v. 83, no. B4, April 1978.

15. Clynch, J., "A Global Positioning System (GPS) Users Seminar", Department of Oceanography, Naval Postgraduate School, Monterey, CA, January 16, 1992.

16. Brown, R. and Hwang, P., *Introduction to Random and Applied Kalman Filtering*, Wiley and Sons, 1992.

17. Kwakernaak, H., and Sivan, R., *Linear Optimal Control Systems*, John Wiley and Sons, 1972.

18. Van Dierendonck, A., McGraw, J., and Brown, R., "Relationship Between Allan Variances and Kalman Filter Parameters", *Proceedings of the Sixteenth Annual Precise Time and Time Interval (PTTI) Applications and Planning Meeting*, pp. 273–293, NASA Goddard Space Flight Center, November 27-29, 1984.

19. Brown, A., "Extended DGPS", *Navigation*, v. 36, no. 3, pp. 265–285, Fall 1989.

20. Langley, R., "The Mathematics of GPS", *GPS World*, August 1991.

21. Lin, C., *Modern Navigation, Guidance, and Control Processing*, Prentice Hall, 1991.

22. Kuechenmeister, D., *A High–Fidelity Non–Linear Simulation of an Autonomous Unmanned Air Vehicle*, Master's Thesis, Naval Postgraduate School, Monterey, CA, September 1993.

23. Gelb, A. and others, *Applied Optimal Estimation*, The MIT Press, 1974.

INITIAL DISTRIBUTION LIST

		No. of Copies
1.	Defense Technical Information Center Cameron Station Alexandria, VA 22304-6145	2
2.	Library, Code 52 Naval Postgraduate School Monterey, CA 93943-5002	2
3.	Dr. Isaac I. Kaminer Department of Aeronautics and Astronautics, Code AA/KA Naval Postgraduate School Monterey, CA 93943-5000	5
4.	Chairman Department of Aeronautics and Astronautics Naval Postgraduate School Monterey, CA 93943-5000	2
5.	John B. Herrington Rt.1 Box 51C Leonardtown, MD 20650	1
6.	James E. Herrington 577 Crosswind Dr. Spicewood, TX 78669	1